

# Uncertainty In Artificial Intelligence

Proceedings of the Thirty-First Conference (2015)

**Edited by**

Marina Meila

Tom Heskes

# Uncertainty in Artificial Intelligence

Proceedings of the Thirty-First Conference (2015)

July 12-16, 2015, Amsterdam, Netherlands

**Edited by**

Marina Meila, University of Washington, USA

Tom Heskes, Radboud University, Netherlands

**General Chair**

Jin Tian, Iowa State University, USA

**Sponsored by**

Artificial Intelligence Journal, Microsoft Research, Facebook Inc.,

Google Inc., Adobe Systems Inc., Baidu Research, Elsevier Labs

**AUAI Press Corvallis, Oregon**

Cover design © Alice Zheng.

Published by AUAI Press for  
Association for Uncertainty in Artificial Intelligence  
<http://auai.org>

Editorial Office:  
P.O. Box 866  
Corvallis, Oregon 97339  
USA

Copyright © 2015 by AUAI Press  
All rights reserved  
Printed in the United States of America

No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means—electronic, mechanical, photocopying, recording, or otherwise—without the prior written permission of the publisher.

ISBN 978-0-9966431-0-8



# Contents

|   |          |
|---|----------|
| Preface   | vii      |
| Organizing Committee  | ix       |
| Acknowledgments   | xi       |
| Sponsors  | xix      |
| Best Paper Awards   | xxi      |
| <b>1 Proceedings</b>  | <b>1</b> |
| Bayesian Optimal Control of Smoothly Parameterized Systems.<br><i>Yasin Abbasi-Yadkori, Csaba Szepesvári</i> . . . . .  | 1        |
| Optimal expert elicitation to reduce interval uncertainty.<br><i>Nadia Ben Abdallah, Sébastien Destercke</i> . . . . .  | 12       |
| Stochastic Integration via Error-Correcting Codes.<br><i>Dimitris Achlioptas, Pei Jiang</i> . . . . .   | 22       |
| Learning the Structure of Sum-Product Networks via an SVD-based Algorithm.<br><i>Tameem Adel, David Balduzzi, Ali Ghodsi</i> . . . . .                              | 32       |
| Robust reconstruction of causal graphical models based on conditional 2-point and 3-point information.<br><i>Séverine Affeldt, Hervé Isambert</i> . . . . .         | 42       |
| Are You Doing What I Think You Are Doing? Criticising Uncertain Agent Models.<br><i>Stefano V. Albrecht, Subramanian Ramamoorthy</i> . . . . .                      | 52       |
| Disciplined Convex Stochastic Programming: A New Framework for Stochastic Optimization.<br><i>Alnur Ali, J. Zico Kolter, Steven Diamond, Stephen Boyd</i> . . . . . | 62       |
| Intelligent Affect: Rational Decision Making for Socially Aligned Agents.<br><i>Nabiha Asghar, Jesse Hoey</i> . . . . .   | 72       |
| Representation Learning for Clustering: A Statistical Framework.<br><i>Hassan Ashtiani, Shai Ben-David</i> . . . . .  | 82       |
| Adversarial Cost-Sensitive Classification.<br><i>Kaiser Asif, Wei Xing, Sima Behpour, Brian D. Ziebart</i> . . . . .  | 92       |
| Geometric Network Comparisons.<br><i>Dena Marie Asta, Cosma Rohilla Shalizi</i> . . . . .   | 102      |
| Learning and Planning with Timing Information in Markov Decision Processes.<br><i>Pierre-Luc Bacon, Borja Balle, Doina Precup</i> . . . . .                         | 111      |
| Parameterizing the Distance Distribution of Undirected Networks.<br><i>Christian Bauckhage, Kristian Kersting, Fabian Hadji</i> . . . . .                           | 121      |
| New Limits for Knowledge Compilation and Applications to Exact Model Counting.<br><i>Paul Beame, Vincent Liew</i> . . . . .   | 131      |
| Hashing-Based Approximate Probabilistic Inference in Hybrid Domains.<br><i>Vaishak Belle, Guy Van den Broeck, Andrea Passerini</i> . . . . .                        | 141      |
| Bayesian Network Learning with Discrete Case-Control Data.<br><i>Giorgos Borboudakis, Ioannis Tsamardinos</i> . . . . .   | 151      |

|  |     |
|--|-----|
| Efficient Algorithms for Bayesian Network Parameter Learning from Incomplete Data.<br><i>Guy Van den Broeck, Karthika Mohan, Arthur Choi, Adnan Darwiche, Judea Pearl</i> . . .          | 161 |
| Bayes Optimal Feature Selection for Supervised Learning with General Performance Measures.<br><i>Saneem Ahmed C.G., Harikrishna Narasimhan, Shivani Agarwal</i> . . . . .                | 171 |
| Visual Causal Feature Learning.<br><i>Krzysztof Chalupka, Pietro Perona, Frederick Eberhardt</i> . . . . .   | 181 |
| Large-Margin Determinantal Point Processes.<br><i>Wei-Lun Chao, Boqing Gong, Kristen Grauman, Fei Sha</i> . . . . .  | 191 |
| Fast Relative-Error Approximation Algorithm for Ridge Regression.<br><i>Shouyuan Chen, Yang Liu, Michael R. Lyu, Irwin King, Shengyu Zhang</i> . . . . .                                 | 201 |
| Selective Greedy Equivalence Search: Finding Optimal Bayesian Networks Using a Polynomial<br>Number of Score Evaluations.<br><i>David Maxwell Chickering, Christopher Meek</i> . . . . . | 211 |
| Stable Spectral Learning Based on Schur Decomposition.<br><i>Nicolò Colombo, Nikos Vlassis</i> . . . . .   | 220 |
| Semi-described and semi-supervised learning with Gaussian processes.<br><i>Andreas Damianou, Neil D. Lawrence</i> . . . . .  | 228 |
| Budget Constraints in Prediction Markets.<br><i>Nikhil Devanur, Miroslav Dudík, Zhiyi Huang, David Pennock</i> . . . . .   | 238 |
| A Probabilistic Logic for Reasoning about Uncertain Temporal Information.<br><i>Dragan Doder, Zoran Ognjanović</i> . . . . .   | 248 |
| Training generative neural networks via Maximum Mean Discrepancy optimization.<br><i>Gintare Karolina Dziugaite, Daniel M. Roy, Zoubin Ghahramani</i> . . . . .                          | 258 |
| Incremental Region Selection for Mini-bucket Elimination Bounds.<br><i>Sholeh Forouzan, Alexander Ihler</i> . . . . .  | 268 |
| Estimating Mutual Information by Local Gaussian Approximation.<br><i>Shuyang Gao, Greg Ver Steeg, Aram Galstyan</i> . . . . .  | 278 |
| Psychophysical Detection Testing with Bayesian Active Learning.<br><i>Jacob R. Gardner, Xinyu Song, Kilian Q. Weinberger, Dennis Barbour, John P.<br/>Cunningham</i> . . . . .           | 286 |
| Locally Conditioned Belief Propagation.<br><i>Thomas Geier, Felix Richter, Susanne Biundo</i> . . . . .  | 296 |
| Discriminative Switching Linear Dynamical Systems applied to Physiological Condition<br>Monitoring.<br><i>Konstantinos Georgatzis, Christopher K. I. Williams</i> . . . . .              | 306 |
| Revisiting Non-Progressive Influence Models: Scalable Influence Maximization in Social<br>Networks.<br><i>Golshan Golnari, Amir Asiaee T., Arindam Banerjee, Zhi-Li Zhang</i> . . . . .  | 316 |
| Scalable Recommendation with Hierarchical Poisson Factorization.<br><i>Prem Gopalan, Jake M. Hofman, David M. Blei</i> . . . . .   | 326 |
| State Sequence Analysis in Hidden Markov Models.<br><i>Yuri Grinberg, Theodore J. Perkins</i> . . . . .  | 336 |
| Multitasking: Optimal Planning for Bandit Superprocesses.<br><i>Dylan Hadfield-Menell, Stuart Russell</i> . . . . .  | 345 |
| Importance Sampling over Sets: A New Probabilistic Inference Scheme.<br><i>Stefan Hadjis, Stefano Ermon</i> . . . . .  | 355 |
| Progressive Abstraction Refinement for Sparse Sampling.<br><i>Jesse Hostetler, Alan Fern, Thomas Dietterich</i> . . . . .  | 365 |
| Zero-Truncated Poisson Tensor Factorization for Massive Binary Tensors.<br><i>Changwei Hu, Piyush Rai, Lawrence Carin</i> . . . . .  | 375 |
| Computing Optimal Bayesian Decisions for Rank Aggregation via MCMC Sampling.<br><i>David Hughes, Kevin Hwang, Lirong Xia</i> . . . . .   | 385 |
| Do-calculus when the True Graph Is Unknown.<br><i>Antti Hyttinen, Frederick Eberhardt, Matti Järvisalo</i> . . . . .   | 395 |

|   |     |
|---|-----|
| Kernel-Based Just-In-Time Learning for Passing Expectation Propagation Messages.<br><i>Wittawat Jitkrittum, Arthur Gretton, Nicolas Heess, S. M. Ali Eslami, Balaji Lakshminarayanan, Dino Sejdinovic, Zoltán Szabó</i> . . . . . | 405 |
| Averaging of Decomposable Graphs by Dynamic Programming and Sampling.<br><i>Kustaa Kangas, Teppo Niinimäki, Mikko Koivisto</i> . . . . .  | 415 |
| Novel Bernstein-like Concentration Inequalities for the Missing Mass.<br><i>Bahman Yari Saeed Khanloo, Gholamreza Haffari</i> . . . . .   | 425 |
| Minimizing Expected Losses in Perturbation Models with Multidimensional Parametric Min-cuts.<br><i>Adrian Kim, Kyomin Jung, Yongsu Lim, Daniel Tarlow, Pushmeet Kohli</i> . . . . .   | 435 |
| Population Empirical Bayes.<br><i>Alp Kucukelbir, David M. Blei</i> . . . . .   | 444 |
| Encoding Markov logic networks in Possibilistic Logic.<br><i>Ondřej Kuželka, Jesse Davis, Steven Schockaert</i> . . . . .   | 454 |
| On the Computability of AIXI.<br><i>Jan Leike, Marcus Hutter</i> . . . . .  | 464 |
| Tracking with ranked signals.<br><i>Tianyang Li, Harsh Pareek, Pradeep Ravikumar, Dhruv Balwada, Kevin Speer</i> . . . . .  | 474 |
| Classification of Sparse and Irregularly Sampled Time Series with Mixtures of Expected Gaussian<br>Kernels and Random Features.<br><i>Steven Cheng-Xian Li, Benjamin Marlin</i> . . . . .   | 484 |
| Complexity of the Exact Solution to the Test Sequencing Problem.<br><i>Wenhao Liu, Ross D. Shachter</i> . . . . .   | 494 |
| Finite-Sample Analysis of Proximal Gradient TD Algorithms.<br><i>Bo Liu, Ji Liu, Mohammad Ghavamzadeh, Sridhar Mahadevan, Marek Petrik</i> . . . . .  | 504 |
| Estimating the Partition Function by Discriminance Sampling.<br><i>Qiang Liu, Jian Peng, Alexander Ihler, John Fisher III</i> . . . . .   | 514 |
| A Finite Population Likelihood Ratio Test of the Sharp Null Hypothesis for Compliers.<br><i>Wen Wei Loh, Thomas S. Richardson</i> . . . . .   | 523 |
| Structure Learning Constrained by Node-Specific Degree Distribution.<br><i>Jianzhu Ma, Feng Zhao, Jinbo Xu</i> . . . . .  | 533 |
| Active Search and Bandits on Graphs using Sigma-Optimality.<br><i>Yifei Ma, Tzu-Kuo Huang, Jeff Schneider</i> . . . . .   | 542 |
| Off-policy learning based on weighted importance sampling with linear computational complexity.<br><i>A. Rupam Mahmood, Richard S. Sutton</i> . . . . .   | 552 |
| Impact of Learning Strategies on the Quality of Bayesian Networks: An Empirical Evaluation.<br><i>Brandon Malone, Matti Järvisalo, Petri Myllymäki</i> . . . . .  | 562 |
| Learning the Structure of Causal Models with Relational and Temporal Dependence.<br><i>Katerina Marazopoulou, Marc Maier, David Jensen</i> . . . . .  | 572 |
| Hamiltonian ABC.<br><i>Edward Meeds, Robert Leenders, Max Welling</i> . . . . .   | 582 |
| (Nearly) Optimal Differentially Private Stochastic Multi-Arm Bandits.<br><i>Nikita Mishra, Abhradeep Thakurta</i> . . . . .   | 592 |
| Equitable Partitions of Concave Free Energies.<br><i>Martin Mladenov, Kristian Kersting</i> . . . . .   | 602 |
| Non-parametric Revenue Optimization for Generalized Second Price auctions..<br><i>Mehryar Mohri, Andrés Muñoz Medina</i> . . . . .  | 612 |
| Polynomial-time algorithm for learning optimal tree-augmented dynamic Bayesian networks.<br><i>José L. Monteiro, Susana Vinga, Alexandra M. Carvalho</i> . . . . .  | 622 |
| Learning and Inference in Tractable Probabilistic Knowledge Bases.<br><i>Mathias Niepert, Pedro Domingos</i> . . . . .  | 632 |
| Multi-Context Models for Reasoning under Partial Knowledge: Generative Process and Inference<br>Grammar.<br><i>Ardavan S. Nobandegani, Ioannis N. Psaromiligkos</i> . . . . .   | 642 |
| Annealed Gradient Descent for Deep Learning.<br><i>Hengyue Pan, Hui Jiang</i> . . . . .   | 652 |

|  |     |
|--|-----|
| Max-Product Belief Propagation for Linear Programming: Applications to Combinatorial Optimization. |     |
| <i>Sejun Park, Jinwoo Shin</i>   | 662 |
| Fast Algorithms for Learning with Long $N$ -grams via Suffix Tree Based Matrix Multiplication.     |     |
| <i>Hristo S. Paskov, John C. Mitchell, Trevor J. Hastie</i>  | 672 |
| A Complete Generalized Adjustment Criterion.   |     |
| <i>Emilija Perković, Johannes Textor, Markus Kalisch, Marloes H. Maathuis</i>                      | 682 |
| Optimal Threshold Control for Energy Arbitrage with Degradable Battery Storage.                    |     |
| <i>Marek Petrik, Xiaojian Wu</i>   | 692 |
| Mesochronal Structure Learning.  |     |
| <i>Sergey Plis, David Danks, Jianyu Yang</i>   | 702 |
| Budgeted Online Collective Inference.  |     |
| <i>Jay Pujara, Ben London, Lise Getoor</i>   | 712 |
| Auxiliary Gibbs Sampling for Inference in Piecewise-Constant Conditional Intensity Models.         |     |
| <i>Zhen Qin, Christian R. Shelton</i>  | 722 |
| Memory-Efficient Symbolic Online Planning for Factored MDPs.                                       |     |
| <i>Aswin Raghavan, Roni Khardon, Prasad Tadepalli, Alan Fern</i>                                   | 732 |
| The Survival Filter: Joint Survival Analysis with a Latent Time Series.                            |     |
| <i>Rajesh Ranganath, Adler Perotte, Noémie Elhadad, David M. Blei</i>                              | 742 |
| Communication Efficient Coresets for Empirical Loss Minimization.                                  |     |
| <i>Sashank J. Reddi, Barnabás Póczos, Alex Smola</i>   | 752 |
| Large-scale randomized-coordinate descent methods with non-separable linear constraints.           |     |
| <i>Sashank J. Reddi, Ahmed Hefny, Carlton Downey, Avinava Dubey, Suvrit Sra</i>                    | 762 |
| An Upper Bound on the Global Optimum in Parameter Estimation.                                      |     |
| <i>Khaled S. Refaat, Adnan Darwiche</i>  | 772 |
| A Markov Game Model for Valuing Player Actions in Ice Hockey.                                      |     |
| <i>Kurt Routley, Oliver Schulte</i>  | 782 |
| Learning Latent Variable Models by Improving Spectral Solutions with Exterior Point Method.        |     |
| <i>Amirreza Shaban, Mehrdad Farajtabar, Bo Xie, Le Song, Byron Boots</i>                           | 792 |
| Missing Data as a Causal and Probabilistic Problem.  |     |
| <i>Ilya Shpitser, Karthika Mohan, Judea Pearl</i>  | 802 |
| Improved Asymmetric Locality Sensitive Hashing (ALSH) for Maximum Inner Product Search (MIPS).     |     |
| <i>Anshumali Shrivastava, Ping Li</i>  | 812 |
| Learning Optimal Chain Graphs with Answer Set Programming.   |     |
| <i>Dag Sonntag, Matti Järvisalo, Jose Peña, Antti Hyttinen</i>                                     | 822 |
| How matroids occur in the context of learning Bayesian network structure.                          |     |
| <i>Milan Studený</i>   | 832 |
| The Long-Run Behavior of Continuous Time Bayesian Networks.  |     |
| <i>Liessman Sturlaugson, John W. Sheppard</i>  | 842 |
| Online Bellman Residual Algorithms with Predictive Error Guarantees.                               |     |
| <i>Wen Sun, J. Andrew Bagnell</i>  | 852 |
| On the Error of Random Fourier Features.   |     |
| <i>Danica J. Sutherland, Jeff Schneider</i>  | 862 |
| Bayesian Structure Learning for Stationary Time Series.  |     |
| <i>Alex Tank, Nicholas J. Foti, Emily B. Fox</i>   | 872 |
| Learning from Pairwise Marginal Independencies.  |     |
| <i>Johannes Textor, Alexander Idelberger, Maciej Liškiewicz</i>                                    | 882 |
| Bethe Projections for Non-Local Inference.   |     |
| <i>Luke Vilnis, David Belanger, Daniel Sheldon, Andrew McCallum</i>                                | 892 |
| A Smart-Dumb/Dumb-Smart Algorithm for Efficient Split-Merge MCMC.                                  |     |
| <i>Wei Wang, Stuart Russell</i>  | 902 |
| Planning under Uncertainty with Weighted State Scenarios.  |     |
| <i>Erwin Walraven, Matthijs T. J. Spaan</i>  | 912 |



|   |     |
|---|-----|
| Generalization Bounds for Transfer Learning under Model Shift.<br><i>Xuezhi Wang, Jeff Schneider</i> . . . . .  | 922 |
| Clustered Sparse Bayesian Learning.<br><i>Yu Wang, David Wipf, Jeong Min Yun, Wei Chen, Ian Wassell</i> . . . . .   | 932 |
| Bethe and Related Pairwise Entropy Approximations.<br><i>Adrian Weller</i> . . . . .  | 942 |
| Efficient Transition Probability Computation for Continuous-Time Branching Processes via<br>Compressed Sensing.<br><i>Jason Xu, Vladimir N. Minin</i> . . . . .           | 952 |
| Extend Transferable Belief Models with Probabilistic Priors.<br><i>Chunlai Zhou, Yuan Feng</i> . . . . .  | 962 |
| Probabilistic Graphical Models Parameter Learning with Transferred Prior and Constraints.<br><i>Yun Zhou, Norman Fenton, Timothy M. Hospedales, Martin Neil</i> . . . . . | 972 |



# Preface

The Conference on Uncertainty in Artificial Intelligence (UAI) is the premier international conference on research related to representation, inference, learning and decision making in the presence of uncertainty within the field of Artificial Intelligence. This volume contains the schedule and abstracts of all papers that were accepted for the 31st UAI Conference, held in Amsterdam, The Netherlands, from July 12 to 16, 2015. Papers appearing in this volume were subjected to a rigorous review process. 291 papers were submitted to the conference (excluding papers that were withdrawn or rejected outright because of potential double submission) and each was peer-reviewed by 3 or more reviewers with the supervision of one Senior Program Committee member. A total of 99 papers were accepted, 28 for oral presentation and 71 for poster presentation, for an acceptance rate of 34%. We are very grateful to the program committee and senior program committee members for their diligent efforts. We are confident that the proceedings, like past UAI conference proceedings, will become an important archival reference for the field.

We are pleased to announce that the Microsoft Best Paper Award is awarded to Vaishak Belle, Guy Van Den Broeck, and Andrea Passerini for their paper “Hashing-based approximate probabilistic inference in hybrid domains”. The Facebook Best Student Paper Award is awarded to Bo Liu (co-authored with Ji Liu, Mohammad Ghavamzadeh, Sridhar Mahadevan, and Marek Petrik) for their paper “Finite-sample analysis of proximal gradient TD algorithms”. The Google Best Student Paper Award is awarded to Wen Sun (co-authored with J. Andrew Bagnell) for their paper “Online Bellman residual algorithms with predictive error guarantees”.

In addition to the presentation of technical papers, we are very pleased to have four distinguished invited speakers at UAI 2015: Peter Bühlmann (ETH Zürich), David MacKay (Cambridge University), David Silver (Google DeepMind), and, as Banquet Speaker, Raphael Slawinski (Mount Royal University). The UAI 2015 tutorials program, chaired by Silja Renooij, consists of four tutorials: “Optimal algorithms for learning Bayesian network structures” by Changhe Yuan, James Cussens, and Brandon Malone, “Computational complexity of Bayesian networks” by Johan Kwisthout and Cassio De Campos, “Belief functions for the working scientist” by Thierry Denoeux and Fabio Cuzzolin, and “Non-parametric causal models” by Robin Evans and Thomas Richardson.

UAI 2015 also hosts three workshops, coordinated by workshops chair Irina Rish: “12th Annual Bayesian Applications Workshop” (John Mark Agosta and Rommel Novaes Carvalho), “StarAI – Statistical Relational AI” (Mathias Niepert, Guy Van den Broeck, Siraam Natarajan, and David Poole) and “Advances in Causal Inference” (Ricardo Silva, Tom Claassen, Robin Evans, Jonas Peters, and Ilya Shpitser).

*Marina Meila and Tom Heskes (Program Co-Chairs)*  
*Jin Tian (General Chair)*



# Organizing Committee

## **General Chair**

Jin Tian, Iowa State University, USA

## **Program Chairs**

Marina Meila, University of Washington, USA

Tom Heskes, Radboud University, Netherlands

## **Tutorials Chair**

Silja Renooij, Universiteit Utrecht, Netherlands

## **Workshops Chair**

Irina Rish, Watson Research Center, USA

## **Proceedings Chair**

Daniel Lowd, University of Oregon, USA

## **Publicity Chair**

Jonas Peters, ETH Zürich, Switzerland

## **Local Arrangements Chair**

Joris Mooij, University of Amsterdam, Netherlands



# Acknowledgments

The success of a conference such as UAI depends greatly on the efforts of many individuals who volunteer their time to provide expert and detailed reviews of submitted papers. In particular, the Program Committee and Senior Program Committee for UAI 2015 were responsible for generating reviews and recommendations for the 291 submissions to the conference. Each submitted paper was reviewed by at least 3 members of the Program Committee. The Senior Program Committee then assessed the individual reviews for each paper, moderated discussion among Program Committee members if needed, and generated meta-reviews and recommendations for the program chairs. We are extremely grateful for the efforts of all of the individuals listed below.

## Senior Program Committee

|                    |  |
|--------------------|--|
| Ayesha Ali         | University of Guelph                               |
| Nina Balcan        | Carnegie Mellon University                         |
| Jeff Bilmes        | University of Washington                           |
| Craig Boutilier    | University of Toronto                              |
| Emma Brunskill     | Carnegie Mellon University                         |
| Kamalika Chaudhuri | University of California, San Diego                |
| Max Chickering     | Microsoft Research                                 |
| Fabio Cuzzolin     | Oxford Brookes University                          |
| Adnan Darwiche     | UCLA   |
| Denver Dash        | Magic Leap   |
| Cassio de Campos   | Queen's University Belfast                         |
| Rina Dechter       | UC-Irvine  |
| Francisco Diez     | UNED   |
| Jennifer Dy        | Northeastern University                            |
| Gal Elidan         | The Hebrew University of Jerusalem                 |
| Helene Fargier     | Institut de Recherche en Informatique de Toulouse  |
| Alexander Ihler    | UC Irvine  |
| Tommi Jaakkola     | MIT  |
| Dominik Janzing    | Max Planck Institute                               |
| Stefanie Jegelka   | University of California, Berkeley                 |
| Helge Langseth     | The Norwegian University of Science and Technology |
| Kathryn Laskey     | George Mason University                            |
| Tze- Yun Leong     | National University of Singapore                   |
| Marloes Maathuis   | ETH Zürich   |
| Chris Meek         | Microsoft Research                                 |
| Claire Monteleoni  | George Washington University                       |
| Remi Munos         | INRIA Lille  |
| Petri Myllymaki    | Helsinki Institute for Information Technology      |
| Ann Nicholson      | Monash University                                  |
| Thomas Nielsen     | Aalborg University                                 |
| David Poole        | University of British Columbia                     |
| Thomas Richardson  | University of Washington                           |
| Prakash Shenoy     | University of Kansas                               |
| Ricardo Silva      | University College London                          |
| Aarti Singh        | Carnegie Mellon University                         |

|                   |  |
|-------------------|--|
| David Sontag      | New York University                            |
| Peter Spirtes     | Carnegie Mellon University                     |
| Claudia Tarantola | University of Pavia                            |
| Raquel Urtasun    | University of Toronto                          |
| Yi Wang           | IHPC, A*STAR                                   |
| Dit-Yan Yeung     | Hong Kong University of Science and Technology |
| Nevin Zhang       | Hong Kong University of Science and Technology |
| Jun Zhu           | Tsinghua University                            |

## Program Committee

|                         |  |
|-------------------------|--|
| Tameem Adel             | Radboud University Nijmegen                                    |
| John Mark Agosta        | Toyota Information Technology Center                           |
| Russell Almond          | Florida State University                                       |
| Christopher Amato       | MIT  |
| Leila Amgoud            | IRIT - Universite Paul Sabatier                                |
| Eyal Amir               | University of Illinois at Urbana-Champaign                     |
| Animashree Anandkumar   | UC Irvine  |
| Alessandro Antonucci    | IDSIA  |
| Cedric Archambeau       | Amazon Berlin  |
| Nimar Arora             | Oracle   |
| Pranjal Awasthi         | Princeton University   |
| Elias Bareinboim        | UCLA   |
| Kim Bauters             | Queen's University of Belfast                                  |
| Nahla Ben Amor          | ISG Tunis  |
| Carlo Berzuni           | University of Manchester                                       |
| Debarun Bhattacharjya   | IBM Research   |
| Bozhena Bidyuk          | Google   |
| Guillaume Bouchard      | Xerox Research Centre Europe                                   |
| Alexandre Bouchard-Cote | UBC  |
| Olivier Buffet          | LORIA-INRIA  |
| Wray Buntine            | Monash University  |
| Cory Butz               | University of Regina   |
| Simon Byrne             | University College London                                      |
| Robert Castelo          | Universitat Pompeu Fabra                                       |
| Hong Chang              | Institute of Computing Technology, Chinese Academy of Sciences |
| Changyou Chen           | Duke University  |
| Ning Chen               | Tsinghua University  |
| Shang-Tse Chen          | Georgia Tech   |
| William Cheung          | Hong Kong Baptist University                                   |
| Arthur Choi             | UCLA   |
| Jaesik Choi             | Ulsan National Institute of Science and Technology             |
| Tianjiao Chu            | University of Pittsburgh                                       |
| Tom Claassen            | Radboud University Nijmegen                                    |
| Giorgio Corani          | IDSIA  |
| Mark Crowley            | Oregon State University  |
| James Cussens           | University of York   |
| Sebastien Destercke     | CNRS   |
| Nicolas Drougard        | ONERA - The French Aerospace Lab                               |
| Marek Druzdel           | University of Pittsburgh                                       |
| Frederick Eberhardt     | Caltech  |
| Zied Eloudi             | ISG Tunis  |
| Stefano Ermon           | Stanford University  |
| Robin Evans             | University of Oxford   |
| M. Julia Flores         | University of Castilla - La Mancha (UCLM)                      |



|                         |  |
|-------------------------|--|
| Aram Galstyan           | Information Sciences Institute             |
| Roman Garnett           | University of Bonn                         |
| Minos Garofalakis       | Technical University of Crete              |
| Phan Giang              | George Mason University                    |
| Bob Givan               | Purdue University ECE                      |
| Lluis Godo              | Artificial Intelligence Research Institute |
| Ali Ghodsi              | University of Waterloo                     |
| Vibhav Gogate           | University of Texas at Dallas              |
| Vincenç Gomez           | Universitat Pompeu Fabra                   |
| Manuel Gomez-Olmedo     | Universidad de Granada                     |
| Christophe Gonzales     | LIP6-UPMC                                  |
| Andrew Gordon Wilson    | Carnegie Mellon University                 |
| Perry Groot             | Radboud University Nijmegen                |
| Roger Grosse            | University of Toronto                      |
| Amit Gruber             | Yahoo!                                     |
| Aritanan Gruber         | University of Sao Paulo                    |
| Yuhong Guo              | Temple University                          |
| Yoni Halpern            | New York University                        |
| Steve Hanneke           | Princeton University                       |
| Tamir Hazan             | University of Haifa                        |
| Philipp Hennig          | Max Planck Institute                       |
| Jesse Hoey              | University of Waterloo                     |
| Arjen Hommersom         | University of Nijmegen                     |
| Antti Honkela           | University of Helsinki                     |
| Bert Huang              | Virginia Tech                              |
| Antti Hyttinen          | University of Helsinki                     |
| Rishabh Iyer            | University of Washington                   |
| David Jensen            | University of Massachusetts Amherst        |
| Abhay Jha               | WalmartLabs                                |
| Alfredo Kalaitzis       | University College London                  |
| Roni Khardon            | Tufts University                           |
| Arto Klami              | University of Helsinki                     |
| Kevin Korb              | Monash University                          |
| Brian Kulis             | Ohio State University                      |
| Akshat Kumar            | IBM Research India                         |
| Balaji Lakshminarayanan | Gatsby/University College London           |
| Jerome Lang             | LAMSADE, CNRS & Universite Paris-Dauphine  |
| Su-In Lee               | University of Washington                   |
| Jan Lemeire             | Vrije Universiteit Brussel                 |
| Philippe Leray          | University of Nantes                       |
| Lei Li                  | Florida International University           |
| Wu-Jun Li               | Nanjing University                         |
| Yujia Li                | University of Toronto                      |
| Yingyu Liang            | Princeton University                       |
| Qihang Lin              | University of Iowa                         |
| Qiang Liu               | UC Irvine                                  |
| Weiru Liu               | Queen's University Belfast                 |
| Yan Liu                 | Southern California                        |
| Ying Liu                | MIT  |
| Samuel Livingstone      | University College London                  |
| Dan Lizotte             | University of Western Ontario              |
| Po-Ling Loh             | University of California, Berkeley         |
| Daniel Lowd             | University of Oregon                       |
| Monia Lupporelli        | University of Bologna                      |
| Manuel Luque            | UNED                                       |

|                       |  |
|-----------------------|--|
| Michael Lyu           | Chinese University of Hong Kong                          |
| Anders Madsen         | Hugin Expert   |
| Malik Magdon-Ismail   | Rensselaer Polytechnic Institute                         |
| Brandon Malone        | Max Planck Institute                                     |
| Radu Marinescu        | IBM Research   |
| Maria Vanina Martinez | University of Oxford                                     |
| Denis Mauá            | University of Sao Paulo                                  |
| Julian McAuley        | UC San Diego   |
| Lukas Meier           | ETH Zürich   |
| Ole Mengshoel         | Carnegie Mellon University                               |
| Ofer Meshi            | Toyota Technological Institute at Chicago                |
| Taneli Mielikainen    | Nokia Research Center Palo Alto                          |
| Brian Milch           | Google   |
| Thomas Minka          | Microsoft Research UK                                    |
| Preetam Nandy         | ETH Zürich   |
| Sriraam Natarajan     | Indiana University                                       |
| Mathias Niepert       | University of Washington                                 |
| William Noble         | University of Washington                                 |
| Nuria Oliver          | Telefonica   |
| Michael Osborne       | Oxford University  |
| David Page            | UW Madison   |
| John Paisley          | Columbia University                                      |
| Xinghao Pan           | University of California, Berkeley                       |
| Jose Pena             | Linköping University                                     |
| Jonas Peters          | ETH Zürich   |
| Marek Petrik          | IBM Research   |
| Kim-Leng Poh          | National University of Singapore                         |
| Leonard Poon          | Hong Kong Institute of Education                         |
| Bob Price             | PARC   |
| David Pynadath        | USC Institute for Creative Technologies                  |
| Erik Quaeghebeur      | Centrum Wiskunde & Informatica                           |
| Piyush Rai            | Duke University  |
| Roland Ramsahai       |  |
| Narges Razavian       | New York University                                      |
| Mark Reid             | Australian National University                           |
| Teemu Roos            | Helsinki Institute for Information Technology            |
| Rafael Rumi           | Almeria University                                       |
| Brian Rutenber        | Charles River Analytics                                  |
| Regis Sabbadin        | INRA   |
| Antonio Salmeron      | Universidad de Almeria                                   |
| Scott Sanner          | NICTA and the Australian National University             |
| Oliver Schulte        | Simon Fraser University                                  |
| Alexander Schwing     | University of Toronto                                    |
| Bart Selman           | Department of Computer Science                           |
| Tomi Silander         | Xerox Research Centre Europe                             |
| Gerardo Simari        | Universidad Nacional del Sur in Bahia Blanca and CONICET |
| Tomas Singliar        | Amazon   |
| Mathieu Sinn          | IBM Research - Ireland                                   |
| Le Song               | Georgia Tech   |
| Fabio Stella          | University of Milan                                      |
| Hang Su               | Tsinghua University                                      |
| L. Enrique Sucar      | INAOE, Mexico  |
| Joe Suzuki            | Osaka University   |
| Vincent Tan           | National University of Singapore                         |
| Danny Tarlow          | Microsoft Research                                       |

|                              |  |
|------------------------------|--|
| Graham Taylor                | University of Guelph                                   |
| Florent Teichteil-Königsbuch | ONERA - The French Aerospace Lab                       |
| Johannes Textor              | Utrecht University                                     |
| Philip Thomas                | University of Massachusetts Amherst                    |
| Yuandong Tian                | Facebook   |
| Ryota Tomioka                | Toyota Technological Institute at Chicago              |
| Ioannis Tsamardinos          | University of Crete                                    |
| Marco Valtorta               | University of South Carolina                           |
| Guy Van den Broeck           | KU Leuven  |
| Twan van Laarhoven           | Radboud University Nijmegen                            |
| Greg Ver Steeg               | Information Sciences Institute                         |
| Jirka Vomlel                 | Institute of Information Theory and Automation         |
| Yevgeniy Vorobeychik         | Vanderbilt University                                  |
| Vladimir Vovk                | Royal Holloway   |
| Thomas Walsh                 | MIT  |
| Chong Wang                   | Carnegie Mellon University                             |
| Shenlong Wang                | University of Toronto                                  |
| Paul Weng                    | Paris 6 University                                     |
| Sinead Williamson            | University of Texas at Austin                          |
| David Wipf                   | Microsoft Research Asia                                |
| Stefan Witwicki              | Ecole Polytechnique Federale de Lausanne               |
| Lirong Xia                   | Rensselaer Polytechnic Institute                       |
| Yang Xiang                   | University of Guelph                                   |
| Minjie Xu                    | Tsinghua University                                    |
| Nan Ye                       | National University of Singapore                       |
| Junming Yin                  | University of Arizona                                  |
| Yaoliang Yu                  | Carnegie Mellon University                             |
| Changhe Yuan                 | City University of New York                            |
| Xiaotong Yuan                | Nanjing University of Information Science & Technology |
| Yifeng Zeng                  | Teesside University                                    |
| Chicheng Zhang               | UC San Diego   |
| Jiji Zhang                   | Lingnan University                                     |
| Kun Zhang                    | MPI for Intelligent Systems                            |
| Ping Zhang                   | IBM Thomas J. Watson Research Center                   |
| Yu Zhang                     | Hong Kong Baptist University                           |
| Yi Zhen                      | Georgia Institute of Technology                        |
| Onno Zoeter                  | Xerox Research Centre Europe                           |

## Additional Reviewers

|                          |  |
|--------------------------|--|
| Bram Arends              | Radboud University Nijmegen                        |
| Aniruddha Basak          | Carnegie Mellon University                         |
| Alan Carlin              | Aptima, Inc.                                       |
| Bryan Chen               | UCLA   |
| Hue Dang                 | Radboud University Nijmegen                        |
| Yawen Fan                | Nanjing University of Posts and Telecommunications |
| José A. Gámez            | University of Castilla – La Mancha                 |
| Zhe Gan                  | Duke University                                    |
| Farhad Ghazvinianzanjani | Radboud University Nijmegen                        |
| Codruta Girlea           | University of Illinois Urbana-Champaign            |
| Nicolas Goix             | Telecom ParisTech                                  |
| Tim Janssen              | Radboud University Nijmegen                        |
| Yacine Jernite           | New York University                                |
| Marcin Kozniewski        | University of Pittsburgh                           |
| Matthijs Lavrijsen       | Radboud University Nijmegen                        |

|                        |  |
|------------------------|--|
| Hoel Le Capitaine      | University of Nantes                               |
| Ritchie Lee            | Carnegie Mellon University                         |
| Miao Liu               | MIT  |
| Pedro Meseguer         | Artificial Intelligence Research Institute         |
| Steffen Michels        | Radboud University Nijmegen                        |
| Trung Nguyen           | Adobe  |
| Gary Overett           | Carnegie Mellon University                         |
| Umut Oztok             | UCLA   |
| Huang Phuong           | Ulsan National Institute of Science and Technology |
| Harmen Prins           | Radboud University Nijmegen                        |
| Wen Pu                 | University of Illinois Urbana-Champaign            |
| José M. Puerta         | University of Castilla – La Mancha                 |
| Deepak Ramachandran    | University of Illinois Urbana-Champaign            |
| Steven Reitsma         | Radboud University Nijmegen                        |
| Yan Shu                | UC San Diego                                       |
| Tom Sterkenburg        | CWI Amsterdam                                      |
| Zhaonan Sun            | IBM  |
| Priya Sundararajan     | Carnegie Mellon University                         |
| Balázs Szörényi        | University of Szeged                               |
| Liang Tang             | Florida International University                   |
| Bas Van Berkel         | Radboud University Nijmegen                        |
| Suzanne Van den Bosch  | Radboud University Nijmegen                        |
| Robbert Van der Gugten | Radboud University Nijmegen                        |
| Fenno Vermeij          | Radboud University Nijmegen                        |
| Amanda Vidal           | Artificial Intelligence Research Institute         |
| Sy Bor Wang            | Adobe  |
| Niklas Weber           | Radboud University Nijmegen                        |
| Shuang Wu              | Shanghai Jiaotong University                       |
| Tong Yu                | Carnegie Mellon University                         |
| Ming Zeng              | Carnegie Mellon University                         |
| Qin Zhou               | Shanghai Jiaotong University                       |
| Xiaoyuan Zhu           | CUNY Queens College                                |

## Additional Acknowledgments

A number of other people have made significant contributions towards making UAI 2015 possible. We acknowledge and thank:

- Laurent Charlin for running the Toronto Publication Matching System.
- Kasper Brink for helping out with CMT.
- Amin Jalali for setting up the conference website.
- Thomas Mensink for serving as the conference webmaster.
- Yali Wan and James McQueen for assisting with the workflow.
- Sara Magliacane, Elles Baaijens and Nicholas Cornia for helping with organizing local arrangements.
- Local volunteers Stephan Bongers, Nicholas Cornia, Hue Dang, Sara Magliacane and Philip Versteeg.
- Kilian Weinberger for setting up the automatic paper formatting checker.
- The following student scholarship volunteers:

|                           |                                     |
|---------------------------|-------------------------------------|
| Amir Asiaee Taheri        | University of Minnesota             |
| Kaiser Asif               | University of Illinois at Chicago   |
| Steven Cheng-Xian Li      | University of Massachusetts Amherst |
| Shuyang Gao               | University of Southern California   |
| Jesse Hostetler           | Oregon State University             |
| Adrian Kim                | Seoul National University           |
| Jan Leike                 | Australian National University      |
| Tianyang Li               | University of Texas at Austin       |
| Wen Wei Loh               | University of Washington            |
| Ben London                | University of Maryland              |
| Aikaterini Marazopoulou   | University of Massachusetts Amherst |
| Nikita Mishra             | University of Chicago               |
| Aswin Nadamuni Raghavan   | Oregon State University             |
| Jay Pujara                | University of Maryland              |
| Wen Sun                   | Carnegie Mellon University          |
| Jason Xu                  | University of Washington            |
| Bahman Yari Saeed Khanloo | Monash University                   |
| Yun Zhou                  | Queen Mary University of London     |



# Sponsors

We gratefully acknowledge the generous support provided by our sponsors, including support for best paper awards and travel scholarships. Without our sponsors' support it would not be feasible to organize a conference such as UAI 2015 without charging much higher registration fees.

## Gold Sponsors



## Silver Sponsors



## Bronze Sponsors







# Best Paper Awards

**Best Paper Award** - sponsored by Microsoft

*Hashing-based approximate probabilistic inference in hybrid domains*

Vaishak Belle, Guy Van Den Broeck, Andrea Passerini

**Facebook Best Student Paper**

*Finite-sample analysis of proximal gradient TD algorithms*

Bo Liu, Ji Liu, Mohammad Ghavamzadeh, Sridhar Mahadevan, Marek Petrik

**Google Best Student Paper**

*Online Bellman residual algorithms with predictive error guarantees*

Wen Sun, J. Andrew Bagnell



# Proceedings

---

# Bayesian Optimal Control of Smoothly Parameterized Systems

---

**Yasin Abbasi-Yadkori**  
Queensland University of Technology

**Csaba Szepesvári**  
University of Alberta

## Abstract

We study Bayesian optimal control of a general class of smoothly parameterized Markov decision problems (MDPs). We propose a *lazy* version of the so-called posterior sampling method, a method that goes back to Thompson and Strens, more recently studied by Osband, Russo and van Roy. While Osband et al. derived a bound on the (Bayesian) regret of this method for undiscounted total cost episodic, finite state and action problems, we consider the continuing, average cost setting with no cardinality restrictions on the state or action spaces. While in the episodic setting, it is natural to switch to a new policy at the episode-ends, in the continuing average cost framework we must introduce switching points explicitly and in a principled fashion, or the regret could grow linearly. Our lazy method introduces these switching points based on monitoring the uncertainty left about the unknown parameter. To develop a suitable and easy-to-compute uncertainty measure, we introduce a new “average local smoothness” condition, which is shown to be satisfied in common examples. Under this, and some additional mild conditions, we derive rate-optimal bounds on the regret of our algorithm. Our general approach allows us to use a single algorithm and a single analysis for a wide range of problems, such as finite MDPs or linear quadratic regulation, both being instances of smoothly parameterized MDPs. The effectiveness of our method is illustrated by means of a simulated example.

## 1 INTRODUCTION

The topic of this paper is Bayesian optimal control, where the problem is to design a policy that achieves optimal performance on the average over control problem instances

that are randomly sampled from a given distribution. This problem naturally arises when the goal is to design a controller for mass-produced systems, where production is imperfect but the errors follow a regular pattern and the goal is to maintain a good average performance over the controlled systems, rather than to achieve good performance even for the system with the largest errors.

In a Bayesian setting, the optimal policy (which exists under appropriate regularity conditions) is history dependent. Given the knowledge of the prior, the transition dynamics and costs, the problem in a Bayesian setting is to find an efficient way to *calculate* the actions that the optimal policy would take given some history. This problem was studied for finite state and action spaces by Asmuth et al. (2009) and Kolter and Ng (2009). Both works propose specific computationally efficient algorithms, which are shown to be  $\epsilon$ -Bayes-optimal with probability  $1 - \delta$  with the exception of  $O(\text{poly}(1/\epsilon))$  many steps, where for both algorithms  $\epsilon$  and  $\delta$  are both part of the input. While Kolter and Ng (2009) suggest to add an exploration bonus to the rewards while using the mean estimates for the transition probabilities and considers a finite horizon setting, Asmuth et al. (2009) consider discounted total rewards and a variant of posterior sampling, originally due to Thompson (1933) and first adapted to reinforcement learning by Strens (2000). More recently, the algorithm of Strens (2000) was revisited by Osband et al. (2013) in the context of episodic, finite MDPs. An attractive feature of posterior sampling is that it requires neither the target accuracy  $\epsilon$ , nor the failure probability  $\delta$  as its inputs. Rather, the guarantee presented by Osband et al. (2013) is that the algorithm’s (Bayesian) regret, i.e., the excess cost due to not following the optimal policy, is bounded by  $\tilde{O}(\sqrt{T})$ <sup>1</sup> both with high probability and in expectation. The reader interested in further algorithms for Bayesian reinforcement learning (including algorithms for infinite state spaces) may consult the papers of Araya-López et al. (2012), Vlassis et al. (2012) and Guez et al. (2013), which together give an excellent overview of the literature.

---

<sup>1</sup> $\tilde{O}(\cdot)$  hides poly-logarithmic factors.

The starting point of our paper is the work of Osband et al. (2013). In particular, just like Osband et al. (2013), we build on the posterior sampling algorithm of Strens (2000), which itself was derived from an algorithm of Thompson (1933) developed for the so-called bandit setting. Unlike Osband et al. (2013) and Strens (2000), we allow the state-action space to be infinite (subject to some regularity conditions discussed later) and we consider the *infinite horizon, continuing, average-cost setting*. As far as we know, ours is the first work deriving (Bayesian) regret bounds for any algorithms of this generality. The major assumption that we make is that *the Markov dynamics is smoothly parameterized in some unknown parameters* with known (local) “smoothness” map such that the posterior concentrates in the metric derived from this map. It is shown that this assumption is met in some common examples, such as finite MDPs, and also in linearly parameterized systems, which encompass, systems with linear dynamics.

Following a proposal of Strens (2000) who also considered the non-episodic setting, the algorithm works in phases: At the beginning of each phase, a policy is computed based on solving the optimal control problem for a random parameter vector drawn from the posterior over the parameter vectors. The algorithm keeps the policy until the parameter uncertainty is reduced by a substantial margin, when a new phase begins and the process is repeated. The idea of ending a phase when uncertainty is reduced by a significant margin goes back at least to the work of Jaksch et al. (2010).

While in the case of episodic problems the issue of how long a policy should be kept does not arise, in a continuing problem with no episodic structure, if policies are changed too often, performance will suffer (see, e.g., Example 1 of Guez et al. (2014)). To address this challenge, for non-episodic problems, Strens (2000) suggested that the lengths of phases should be adjusted to the “planning horizon” (Strens, 2000), which however, is ill-defined for the average cost setting that we consider in this paper. A major contribution of this work is that we show how the smoothness map can be used to derive the length of the phases.

In a recent and independent work, Osband and Van Roy (2014) propose and analyze a similar algorithm for episodic problems. Also, Gopalan and Mannor (2015) show a frequentist analysis of Thompson sampling for finite MDP problems.

The continuing setting is very common in practice; this setting is the most natural for controlled mechanical systems (e.g., CD/DVD drive control, control of manufacturing robots), or for process optimization (e.g., controlling a queuing system, resource management), where “resets” are rare or unnatural.

Under some additional technical conditions, we show that the expected (Bayesian) regret of our algorithm is  $\tilde{O}(\sqrt{T})$

where  $T$  is the number of time steps and  $\Sigma_T$  is controlled by the precision with which the optimal control problems are solved, thus providing an explicit bound on the cost of using imprecise calculations. In summary, the main result of the paper shows that near-optimal Bayesian optimal control is possible for a wide range of problems as long as we can efficiently sample from the parameter posteriors, the length of phases for how long the same policy is followed is carefully controlled and if we can efficiently solve the arising classical optimal control problems. Due to the lack of space, the proofs of some of our claims are given in the supplementary material.

We emphasize two contributions: (1) the invention of a class of systems which unifies many previous approaches, and permits an elegant proof. (2) the introduction of a Concentrating Posterior assumption which significantly shortens our proof compared to previous proofs and improves the bound, as we avoid the use of measure concentration arguments which were always used previously.

## 2 PROBLEM SETTING

We consider problems when the transition dynamics is parameterized with a matrix  $\Theta_* \in \mathbb{R}^{m \times n}$ , which is *randomly chosen* at time 0 (before the interaction with the learner starts) from a known prior  $P_0$  with support  $\mathcal{S} \subset \mathbb{R}^{m \times n}$ . Let  $P_t$  denote the posterior of  $\Theta_*$  at time  $t$  based on  $x_1, a_1, \dots, a_{t-1}, x_t$ . Let  $\mathcal{X} \subset \mathbb{R}^n$  be the state space and  $\mathcal{A} \subset \mathbb{R}^d$  be the action space,  $x_t \in \mathcal{X}$  be the state at time  $t$  and  $a_t \in \mathcal{A}$  be the action at time  $t$ , which is chosen based on  $x_1, a_1, \dots, a_{t-1}, x_t$ . It is assumed that  $x_1$  is sampled from a fixed distribution (although, it should become clear later that this assumption is not necessary). For  $M \succeq 0$  positive semidefinite, define  $\|\Theta\|_M^2 = \|\Theta^\top M \Theta\|_2$ , where  $\|\cdot\|_2$  denotes the spectral norm of matrices (later we will drop the subindex 2). The set of positive semidefinite  $m \times m$  matrices will be denoted by  $\mathbb{S}^+(m)$ . Our main assumption concerning the transition law is as follows:

### Assumption A1 (Smoothly Parameterized Dynamics)

The next state satisfies  $x_{t+1} = f(x_t, a_t, \Theta_*, z_{t+1})$ , where  $z_{t+1} \sim U[0, 1]$  is independent of the past and  $\Theta_*$ . Further, there exists a (known) map  $M : \mathcal{X} \times \mathcal{A} \rightarrow \mathbb{S}^+(m)$  such that for any  $\Theta, \Theta' \in \mathcal{S}$ , if  $y = f(x, a, \Theta, z)$ ,  $y' = f(x, a, \Theta', z)$  with  $z \sim U[0, 1]$ , then  $\mathbb{E}[\|y - y'\|] \leq \|\Theta - \Theta'\|_{M(x,a)}$ .

The first part of the assumption just states that given  $\Theta_*$ , the dynamics is Markovian with state  $x_t$ , while the second part demands that small changes in the parameter lead to small changes in the next state. The assumption that the map  $M$  is “known” makes it possible to use  $M$  in the design of our algorithms.

Our next assumption connects the concentration of the posterior with  $M$ :

**Assumption A2 (Concentrating Posterior)** Let  $\tilde{\mathcal{F}}_t = \sigma(x_1, a_1, \dots, a_{t-1}, x_t)$  be the  $\sigma$ -algebra generated by observations up to time  $t$ ,  $V_t = V + \sum_{s=1}^{t-1} M(x_s, a_s)$ , where  $V$  is an  $m \times m$  positive definite matrix. Then, there exists a positive constant  $C$  such that for any  $t \geq 1$ , for some  $\tilde{\mathcal{F}}_t$ -measurable random variable  $\hat{\Theta}_t$ , letting  $\Theta'_t \sim P_t$  it holds that  $\max \left\{ \mathbb{E} \left[ \|\Theta'_t - \hat{\Theta}_t\|_{V_t}^2 \right], \mathbb{E} \left[ \|\Theta_* - \hat{\Theta}_t\|_{V_t}^2 \right] \right\} \leq C$ .

The idea here is that  $\hat{\Theta}_t$  is an estimate of  $\Theta_*$  based on past information available at time  $t$ , such as a maximum a posteriori (MAP) estimate (note that this estimate will *not* be needed by our algorithm). Since  $V_t$  is increasing at a linear rate, the assumption requires that  $\hat{\Theta}_t$  converges to  $\Theta$  at an  $O(1/\sqrt{t})$  rate. When  $\Theta = \Theta_*$ , this means that  $\hat{\Theta}_t$  should converge to  $\Theta_*$  at this rate, which is indeed what we expect. When  $\Theta = \Theta'_t$ , again, we expect this to be true since  $\Theta'_t$  is expected to be in the  $O(1/\sqrt{t})$  vicinity of  $\Theta_*$ . Note how this assumption connects  $M$  with the behavior of the posterior. One novelty of our analysis, as compared to that of Osband et al. (2013), is that while Osband et al. relies on measure-concentration, we require only the above (weaker) “variance concentration”. We will show explicit examples where this variance term is easy to control using a direct calculation. Since we avoid measure-concentration, our analysis has the potential to give much tighter regret bounds for the Bayesian setting than available previously, though the study of this remains for future work. The examples we deal with include finite MDPs (where the state is represented by unit vectors) and systems with linear dynamics (i.e., when  $x_{t+1} = Ax_t + Ba_t + w_{t+1}$ , where  $w_{t+1} \sim p_w(\cdot|x_t, a_t)$ ), amongst others. Explicit expressions for the map  $M$  will be given in Section 6 for these systems. In general, for systems with additive noise, finding  $M$  essentially reduces to finding a suitable local linearization of the system’s dynamics.

The problem we study is to design a controller (also known as a policy) that at every time step  $t$ , based on past states  $x_1, \dots, x_t$  and actions  $a_1, \dots, a_{t-1}$ , selects an action  $a_t$  so as to minimize the expected long-run average loss  $\mathbb{E} \left[ \limsup_{n \rightarrow \infty} \frac{1}{n} \sum_{t=1}^n \ell(x_t, a_t) \right]$ . We consider any noise distribution and any loss function  $\ell$  as long as a boundedness assumption on the variance and a smoothness assumption on the *value function* are satisfied (see Assumptions A2 and A3-ii below). It is important to note that we allow  $\ell$  to be a nonlinear function of the last state-action pair, i.e., the framework allows one to go significantly beyond the scope of linear quadratic control as many nonlinear control problems can be transformed into a linear form (but with a nonlinear loss function) using the so-called dynamic feedback linearization techniques (Isidori, 1995).

To measure the performance of an algorithm, we use the (expected) regret  $R_T$ :  $R_T = \mathbb{E} \left[ \sum_{t=1}^T (\ell(x_t, a_t) - J(\Theta_*)) \right]$ . Here,  $(x_t, a_t)_{t=1}^T$  de-

notes the state-action trajectory and  $J(\Theta_*)$  is the average loss of the optimal policy given (random) parameter  $\Theta_*$ . The slower the regret grows, the closer is the performance to that of an optimal policy. If the growth rate of  $R_T$  is sublinear ( $R_T = o(T)$ ), the average loss per time step will converge to the optimal average loss as  $T$  gets large and in this sense we can say that the algorithm is asymptotically-optimal. Our main result shows that, under some conditions, the construction of such asymptotically-optimal policies can be reduced to the ability of efficiently sampling from the posterior of  $\Theta_*$  and being able to solve classical (non-Bayesian) optimal-control problems. Furthermore, our main result also implies that  $R_T = \tilde{O}(\sqrt{T})$ .

### 3 THE LAZY PSRL ALGORITHM

Our algorithm is an instance of the posterior sampling reinforcement learning (PSRL) (Osband et al., 2013). As explained beforehand, this algorithm is based on the work on Thompson (1933) and was proposed by Strens (2000). To emphasize that the algorithm keeps the current policy for a while, we call it LAZY PSRL. Our contribution is to suggest a specific schedule for updating the policy. The pseudocode of the algorithm is shown in Figure 1.

Recall that  $P_0$  denotes the prior distribution of the parameter matrix  $\Theta_*$ . Let  $P_t$  denote the posterior of  $\Theta_*$  at time  $t$  based on  $x_1, a_1, \dots, a_{t-1}, x_t$  and  $\tau_t < t$  the last round when the algorithm chose a new policy. Further, let  $V_t = V + \sum_{s=1}^{t-1} M(x_s, a_s)$ , where  $V$  is some fixed,  $m \times m$  positive definite matrix. Let  $G$  be a constant that controls the replanning frequency. Then, at time  $t$ , Lazy PSRL sets  $\tilde{\Theta}_t = \tilde{\Theta}_{t-1}$  unless  $\det(V_t) > G \det(V_{\tau_t})$  in which case it chooses  $\tilde{\Theta}_t$  from the posterior  $P_t$ :  $\tilde{\Theta}_t \sim P_t$ . The action taken at time step  $t$  is a near-optimal action for the system whose transition dynamics is specified by  $\tilde{\Theta}_t$ . We assume that a subroutine,  $\pi^*$ , taking the current state  $x_t$  and the parameter  $\tilde{\Theta}_t$  is available to calculate such an action. The inexact nature of calculating a near-optimal action will also be taken in our analysis.

### 4 RESULTS FOR BOUNDED STATE- AND FEATURE-SPACES

In this section, we study problems with a bounded state space. In particular, the number of states might be infinite, but we assume that the norm of the state vector is bounded by a constant. Before stating our main result, we state some extra assumptions.

Our first extra assumption concerns the existence of “regular” solutions to the average cost optimality equations (ACOE), an assumption which is usually thought to be mild in the context of average-cost problems:

```

Inputs:  $P_0$ , the prior distribution of  $\Theta_*$ ,  $V$ ,  $G$ .
 $V_{\text{last}} \leftarrow V, V_0 \leftarrow V$ .
for  $t \leftarrow 1, 2, \dots$  do
  if  $\det(V_t) > G \det(V_{\text{last}})$  then
    Sample  $\tilde{\Theta}_t \sim P_t$ .
     $V_{\text{last}} \leftarrow V_t$ .
  else
     $\tilde{\Theta}_t \leftarrow \tilde{\Theta}_{t-1}$ .
  end if
  Calculate near-optimal action  $a_t \leftarrow \pi^*(x_t, \tilde{\Theta}_t)$ .
  Execute action  $a_t$  and observe the new state  $x_{t+1}$ .
  Update  $P_t$  with  $(x_t, a_t, x_{t+1})$  to obtain  $P_{t+1}$ .
  Update  $V_{t+1} \leftarrow V_t + M(x_t, a_t)$ .
end for

```

Figure 1: Lazy PSRL for smoothly parameterized control problems

**Assumption A3 (Existence of Regular ACOE Solutions)**  
The following hold:

- (i) There exists  $H > 0$  such that for any  $\Theta \in \mathcal{S}$ , there exist a scalar  $J(\Theta)$  and a function  $h(\cdot, \Theta) : \mathcal{X} \rightarrow [0, H]$  that satisfy the average cost optimality equation (ACOE): for any  $x \in \mathcal{X}$ ,

$$J(\Theta) + h(x, \Theta) = \min_{a \in \mathcal{A}} \left\{ \ell(x, a) + \int h(y, \Theta) p(dy | x, a, \Theta) \right\}, \quad (1)$$

where  $p(\cdot | x, a, \Theta)$  is the next-state distribution given state  $x$ , action  $a$  and parameter  $\Theta$ .

- (ii) There exists  $B > 0$  such that for all  $\Theta \in \mathcal{S}$ , and for all  $x, x' \in \mathcal{X}$ ,  $|h(x, \Theta) - h(x', \Theta)| \leq B \|x - x'\|$ .

With a slight abuse of the concepts, we will call the quantity  $J(\Theta)$  the average loss of the optimal policy, while function  $h(\cdot, \Theta)$  will be called the value function (for the system with parameter  $\Theta$ ). The review paper by Arapostathis et al. (1993) gives a number of sufficient (and sometimes necessary) conditions that guarantee that a solution to ACOE exists. Lipschitz continuity usually follows from that of the transition dynamics and the losses.

Let us now discuss the condition that  $h$  should have a bounded range. A uniform lower bound on  $h$  follows, for example if the immediate cost function  $\ell$  is lower bounded. Then, if the state space is bounded, uniform boundedness of the functions  $h(\cdot, \Theta)$  follows from their uniform Lipschitzness:

**Proposition 1.** *Assume that the value function  $h(\cdot, \Theta)$  is bounded from below ( $\inf_x h(x, \Theta) > -\infty$ ) and is  $B$ -Lipschitz. Then, if the diameter of the state space is*

*bounded by  $X$  (i.e.,  $\sup_{x, x' \in \mathcal{X}} \|x - x'\| \leq X$ ) then there exists a solution  $h'(\cdot, \Theta)$  to (1) such that the range of  $h$  is included in  $[0, BX]$ .*

Finally, we assume that the map  $M : \mathcal{X} \times \mathcal{A} \rightarrow \mathbb{S}^+(m)$  is bounded:

**Assumption A4 (Boundedness)** There exist  $\Phi > 0$  such that for all  $x \in \mathcal{X}$  and  $a \in \mathcal{A}$ ,  $\text{trace}(M(x, a)) \leq \Phi^2$ .

This assumption may be strong. In the next section we discuss an extension of the result of this section to the case when this assumption is not met.

The main theorem of this section bounds the regret of Lazy PSRL under the assumptions mentioned so far. In this result, we allow  $\pi^*$  to return a  $\sigma_t$ -suboptimal action, where  $\sigma_t > 0$ . By this, we mean that the action  $a_t$  satisfies

$$\ell(x_t, a_t) + \int h(y, \tilde{\Theta}_t) p(dy | x_t, a_t, \tilde{\Theta}_t) \leq \min_{a \in \mathcal{A}} \left\{ \ell(x_t, a) + \int h(y, \tilde{\Theta}_t) p(dy | x_t, a, \tilde{\Theta}_t) \right\} + \sigma_t. \quad (2)$$

One can control the suboptimality error in terms of the error of an approximate solution to the Bellman equation and the error of the subroutine that finds an action that minimizes the obtained approximate action values.

**Theorem 2.** *Assume that A1–A4 hold for some values of  $C, B, X, \Phi > 0$ . Consider Lazy PSRL where in time step  $t$ , the action chosen is  $\sigma_t$ -suboptimal. Then, for any time  $T$ , the regret of Lazy PSRL satisfies  $R_T = \tilde{O}(\sqrt{T}) + \Sigma_T$ , where  $\Sigma_T = \sum_{t=1}^T \mathbb{E}[\sigma_t]$  and the constant hidden by  $\tilde{O}(\cdot)$  depends on  $V, C, B, X, G$  and  $\Phi$ .*

In particular, the theorem implies that Lazy PSRL is asymptotically optimal as long as  $\sum_{t=1}^T \mathbb{E}[\sigma_t] = o(T)$  and it is  $O(\epsilon)$ -optimal if  $\mathbb{E}[\sigma_t] \leq \epsilon$ . The fact that the regret is bounded by the sum of suboptimality factors in solving Bellman equation is not trivial. Indeed, as actions have long term effects and we have a closed-loop system, one might suspect that the regret could blow up as a function of these errors. In this respect, the significance of our theorem is that the learner need not worry too much about each planning subproblem as the overall effect is only additive.

Due to lack of space, the proof, which combines the proof techniques of Osband et al. (2013) with that of Abbasi-Yadkori and Szepesvári (2011) in a novel fashion, is presented in the appendix.

## 5 FORCEFULLY STABILIZED SYSTEMS

For some applications, such as robotics, where the state can grow unbounded, the boundedness assumption (Assumption A4) is rather problematic. For such systems, it is common to use a stabilizing controller  $\pi_{\text{stab}}$  that is automatically turned on and is kept on as long as the state vector is

“large”. The stabilizing controller, however, is usually expensive (uses lots of energy), as it is designed to be robust so that it is guaranteed to drive back the state to the safe region for all possible systems under consideration. Hence a good controller should avoid relying on the stabilizing controller.

In this section, we will replace Assumption A4 with an assumption that a stabilizing controller is available. We will use this controller to override the actions coming from our algorithm as soon as the state leaves the (bounded) safe region  $\mathcal{R} \subset \mathbb{R}^n$  until it returns to it. The corresponding pseudocode is shown in Figure 2.

**Inputs:**  $P_0$ , the prior distribution of  $\Theta_*$ ,  $V$ , the safe region  $\mathcal{R} \subset \mathbb{R}^n$ .  
Initialize Lazy PSRL with  $P_0$  and  $V$ ,  $x_1$ .  
**for**  $t = 1, 2, \dots$  **do**  
  **if**  $x_t \in \mathcal{R}$  **then**  
    Get action  $a_t$  from Lazy PSRL  
  **else**  
    Get action  $a_t$  from  $\pi_{\text{stab}}$   
  **end if**  
  Execute action  $a_t$  and observe the new state  $x_{t+1}$ .  
  Feed  $a_t$  and  $x_{t+1}$  to Lazy PSRL.  
**end for**

Figure 2: Stabilized Lazy PSRL

We assume that the stabilizing controller is effective in the following sense:

**Assumption A5 (Effective Stabilizing Controller)** There exists  $\Phi > 0$  such that the following holds: Pick any  $x \in \mathcal{R}$ ,  $a \in \mathcal{A}$  and let  $x'_1, a'_1, x'_2, a'_2, \dots$  be the sequence of state-action pairs obtained when from time step two the Markovian stabilizing controller  $\pi_{\text{stab}}$  is applied to the controlled system whose dynamics is given by  $\Theta \in \mathcal{S}$ :  $x'_1 = x$ ,  $a'_1 = a$ ,  $x'_{t+1} \sim p(\cdot|x'_t, a'_t, \Theta)$ ,  $a'_{t+1} \sim \pi_{\text{stab}}(\cdot|x'_t)$ . Then,  $\mathbb{E}[\text{trace}(M(x'_t, a'_t))] \leq \Phi^2$  for any  $t \geq 1$ , where  $M : \mathcal{X} \times \mathcal{A} \rightarrow \mathbb{S}^+(m)$  is the map of Assumption A1 underlying  $\{p(\cdot|x, a, \Theta)\}$ .

The assumption is reasonable as it only requires that the trace of  $M(x'_t, a'_t)$  is bounded *in expectation*. Thus, large spikes, that no controller may prevent, can exist as long as they happen with a sufficiently low probability.

The next theorem shows that Stabilized Lazy PSRL is near Bayes-optimal for the system  $p'$  obtained from  $p$  by overwriting the action  $a$  by the action  $\pi_{\text{stab}}(x)$  if  $x$  is outside of the safe region  $\mathcal{R} \subset \mathbb{R}^n$ :

$$p'(dy|x, a, \Theta) = \begin{cases} p(dy|x, a, \Theta), & \text{if } x \in \mathcal{R}; \\ p(dy|x, \pi_{\text{stab}}(x), \Theta), & \text{otherwise.} \end{cases}$$

**Theorem 3.** Consider a parameterized system with the transition probability kernel family  $\{p(\cdot|x, a, \Theta)\}_{\Theta \in \mathcal{S}}$  and let  $\pi_{\text{stab}} : \mathcal{X} \rightarrow \mathcal{A}$  be a deterministic Markovian controller. Let the smooth parameterization Assumption A1 hold for  $\{p(\cdot|x, a, \Theta)\}$ , the ACOE solution regularity Assumption A3 hold for  $\{p'(\cdot|x, a, \Theta)\}$ . Consider running the Stabilized Lazy PSRL algorithm of Figure 2 on  $p(\cdot|x, a, \Theta_*)$  and let the concentration Assumption A2 hold along the trajectory obtained. Then, if in addition Assumption A5 holds then the regret of Stabilized Lazy PSRL against the Bayesian optimal controller of  $\{p'(\cdot|x, a, \Theta)\}_{\Theta}$  with prior  $P_0$  and immediate cost  $\ell$  satisfies  $R_T = \tilde{O}(\sqrt{T}) + \Sigma_T$ , where  $\Sigma_T = \sum_{t=1}^T \mathbb{E}[\mathbf{1}\{x_t \in \mathcal{R}\} \sigma_t]$  and  $\sigma_t$  is the suboptimality of the action computed by Lazy PSRL at time step  $t$ .

If the optimal controller  $\pi^*$  for  $p$  does not excite the condition that turns on the stabilizing controller, then this controller is also optimal for  $p'$ . In this case, Stabilized Lazy PSRL will have the same regret against  $\pi^*$  than what it has against the optimal controller of  $p'$  and the theorem implies that it will achieve sublinear regret in the original system, as long as  $\Sigma_T$  is sublinear.

## 6 EXAMPLES

The purpose of this section is to illustrate the results obtained. In particular, we will consider applying the results to finite MDPs and linearly parameterized controlled systems and show that for these cases all the assumptions can be satisfied and Lazy PSRL can achieve a low expected regret. We believe that our results will be applicable to many more settings, such as hybrid discrete-continuous systems where the discrete states control which continuous dynamics is used.

### 6.1 Finite MDPs

Consider an MDP problem with finite state and action spaces. Let the state space be  $\mathcal{X} = \{1, 2, \dots, n\}$  and the action space be  $\mathcal{A} = \{1, 2, \dots, d\}$ . We represent the state variable by an  $n$ -dimensional binary vector  $x_t$  that has only one non-zero element at the current state and will write the dynamics in the form  $x_{t+1} = \Theta_* \varphi(x_t, a_t) + \eta_t$ , where  $\Theta_*$  will collect the transition matrices into a single big matrix and  $\eta_t$  is a “Markov noise”. The feature map,  $\varphi : \mathcal{X} \times \mathcal{A} \rightarrow \mathbb{R}^{nd}$  and the parameter matrix are defined as follows: for  $1 \leq k \leq nd$ ,

$$\varphi_k(x, a) = \begin{cases} 1, & \text{if } k = (a-1)n + x; \\ 0, & \text{otherwise,} \end{cases} \quad \Theta_* = \begin{pmatrix} \Theta_*^{(1)} \\ \Theta_*^{(2)} \\ \vdots \\ \Theta_*^{(d)} \end{pmatrix}.$$



Let  $s \in [n]$  be a state and  $a \in [d]$  be an action. The  $s$ th row of matrix  $\Theta_*^{(a)}$  is a distribution over the state space that shows the transition probabilities when we take action  $a$  in state  $s$ . Thus, any row of  $\Theta_*^{(a)}$  sums to one and  $\mathbb{E}[x_{t+1}|x_t, a_t] = \Theta_*^\top \varphi(x_t, a_t)$ .

An appropriate prior for each row is a Dirichlet distribution. Let  $\alpha_1, \dots, \alpha_n$  be positive numbers and let  $V' = \text{diag}(\alpha_1, \dots, \alpha_n)$ . Then  $V = \text{diag}(V', \dots, V') \in \mathbb{R}^{nd \times nd}$  is our ‘‘smoother’’. Let the prior for the  $s$ th row of  $\Theta_*^{(a)}$  be the Dirichlet distribution with parameters  $(\alpha_1, \dots, \alpha_n)$ :  $(P_0)_{s,:} = D(\alpha_1, \dots, \alpha_n)$ . At time  $t$ , the posterior has the form

$$(P_t)_{s,:} = D(\alpha_1 + c_t(s, a, 1), \dots, \alpha_n + c_t(s, a, n)),$$

where  $c_t(s, a, s')$  is the number of observed transitions to state  $s'$  after taking action  $a$  in state  $s$  during the first  $t$  time steps. Matrix  $V_t$  is a diagonal matrix with diagonal elements depending only on the number of times a state-action pair is observed. In particular,

$$(V_t)_{n(a-1)+s, n(a-1)+s} = \sum_{s'} (\alpha_{s'} + c_t(s, a, s')).$$

Vector  $\widehat{\Theta}_{t,(:,s')}$  is an  $nd$ -dimensional vector and its elements show the empirical frequency of transition to state  $s'$  from different state-action pairs. The mean of distribution  $(P_t)_{s,:}$  is the vector  $\widehat{\Theta}_{t,(n(a-1)+s,:)}$  where

$$\widehat{\Theta}_{t,(n(a-1)+s,:)} = \frac{\alpha_{s'} + c_t(s, a, s')}{\sum_{s''} (\alpha_{s''} + c_t(s, a, s''))}.$$

We now show that matrix-valued map  $M$  can be chosen to be  $M(x, a) = (\sqrt{2}/2)\mathbb{I}$ :

**Proposition 4.** *The above choice makes Assumptions A1 and A2 satisfied.*

*Proof.* Let us first show that Assumption A1 holds. Because  $\mathbb{E}[y|x, a] = \Theta^\top \varphi(x, a)$ ,  $\mathbb{E}[y'|x, a] = \Theta'^\top \varphi(x, a)$ , and  $y$  and  $y'$  have only one non-zero element,

$$\begin{aligned} \mathbb{E}[\|y - y'\|] &= \sqrt{2}\mathbb{P}(y \neq y') = \sqrt{2}(1 - \mathbb{P}(y = y')) \\ &= \sqrt{2}\left(1 - \Theta_{(x,a),:}^\top \Theta'_{(x,a),:}\right) \\ &= \frac{\sqrt{2}}{2} \left\| \Theta_{(x,a),:} - \Theta'_{(x,a),:} \right\|^2, \end{aligned}$$

where the last step holds because each row of  $\Theta$  and  $\Theta'$  sum to one.

Let us now prove that Assumption A2 holds: Let  $N = (\Theta_* - \widehat{\Theta}_t)^\top$ ,  $\alpha_{s,a,s'} = \alpha_{s'} + c_t(s, a, s')$  and  $\bar{\alpha}_{s,a} = \sum_{s'} \alpha_{s,a,s'} = V_{t,(n(a-1)+s, n(a-1)+s)}$ . Let  $\|\cdot\|_F$  denote the

Frobenius norm. We have that

$$\begin{aligned} \mathbb{E}\left[\left\|\|NV_t^{1/2}\|^2 \mid \mathcal{F}_t\right.\right] &\leq \mathbb{E}\left[\left\|\|NV_t^{1/2}\|_F^2 \mid \mathcal{F}_t\right.\right] \\ &= \mathbb{E}\left[\sum_{s,a} V_{t,(n(a-1)+s, n(a-1)+s)} \sum_{s'} N_{s',n(a-1)+s}^2 \mid \mathcal{F}_t\right] \\ &= \sum_{s,a} \bar{\alpha}_{s,a} \sum_{s'} \mathbb{E}\left[N_{s',n(a-1)+s}^2 \mid \mathcal{F}_t\right]. \end{aligned}$$

Because each row of  $\Theta_*$  has a Dirichlet distribution and rows of  $\widehat{\Theta}_t$  are means of these distributions,  $\mathbb{E}\left[N_{s',n(a-1)+s}^2 \mid \mathcal{F}_t\right]$  is simply the variance of the corresponding Dirichlet variable. Thus,

$$\begin{aligned} \mathbb{E}\left[\left\|\|NV_t^{1/2}\|^2 \mid \mathcal{F}_t\right.\right] &\leq \sum_{s,a} \sum_{s'} \frac{\bar{\alpha}_{s,a} \alpha_{s,a,s'} (\bar{\alpha}_{s,a} - \alpha_{s,a,s'})}{\bar{\alpha}_{s,a}^2 (1 + \bar{\alpha}_{s,a})} \\ &\leq n^2 d. \end{aligned}$$

□

An immediate corollary of this is that Lazy PSRL will enjoy low regret in finite MDPs:

**Corollary 5.** *Consider Lazy PSRL applied to a finite MDP with  $n$  states,  $d$  actions with  $M$  as above, and a Dirichlet prior as specified above. Assume that the set  $\mathcal{S}$  system parameters under which Assumption A3 is satisfied is a measurable set with positive Lebesgue measure. Suppose that at time step  $t$ , the action chosen is  $\sigma_t$ -suboptimal. Then, for any time  $T$ , the regret of Lazy PSRL satisfies  $R_T = \widetilde{O}(\sqrt{T}) + \Sigma_T$ .*

*Proof.* The boundedness condition (Assumption A4) trivially holds, Assumption A3 holds by assumption, while Proposition 4 shows that the remaining two assumptions of Theorem 2 are satisfied. □

## 6.2 Linearly Parametrized Problems with Gaussian Noise

Next, we consider linearly parametrized problems with Gaussian noise:

$$x_{t+1} = \Theta_*^\top \varphi(x_t, a_t) + w_{t+1}, \quad (3)$$

where  $w_{t+1}$  is a zero-mean normal random variable. The nonlinear dynamics shown in (3) shares similarities to, but allows significantly greater generality than the Linear Quadratic (LQ) problem considered by Abbasi-Yadkori and Szepesvári (2011). In particular, in the LQ problem,  $\Theta_*^\top = (A_*, B_*)$  and  $\varphi(x_t, a_t)^\top = (x_t^\top, a_t^\top)$ . (However, Abbasi-Yadkori and Szepesvári (2011) assume only that the noise is subgaussian.)

Next, we describe a conjugate prior under the assumption that the noise is Gaussian with a known covariance matrix. Without loss of generality, we assume that

$\mathbb{E} [w_{t+1} w_{t+1}^\top | \mathcal{F}_t] = I$ . A conjugate prior is appealing as the posterior has a compact representation that allows for computationally efficient sampling methods. Assume that the columns of matrix  $\Theta_*$  are independently sampled from the following prior: for  $i = 1 \dots n$ ,

$$P_0(\Theta_{*,(:,i)}) \propto \exp\left(\Theta_{*,(:,i)}^\top V \Theta_{*,(:,i)}\right) \mathbf{1}\{\Theta_{*,(:,i)} \in \mathcal{S}\}$$

and  $\mathcal{S}$  is the set of system parameters under which Assumption A3 is satisfied, which is assumed to be a measurable set with positive Lebesgue measure. Then, by Bayes' rule, the posterior for column  $i$  of  $\Theta_*$ ,  $P_t(\Theta_{*,(:,i)})$ , is proportional to

$$e^{(-0.5(\Theta_{*,(:,i)} - \hat{\Theta}_{t,(:,i)})^\top V_t (\Theta_{*,(:,i)} - \hat{\Theta}_{t,(:,i)}))} \mathbf{1}\{\Theta_{*,(:,i)} \in \mathcal{S}\}.$$

We now show an appropriate choice for  $M$  (which should not be surprising):

**Proposition 6.** *With the choice  $M(x, a) = \varphi(x, a)\varphi(x, a)^\top$ , Assumptions A1 and A2 are satisfied.*

Note that this choice is essentially the same as in Proposition 4.

*Proof.* Let us first show that Assumption A1 holds. Because  $y = \Theta^\top \varphi(x, a) + w$ ,  $y' = \Theta'^\top \varphi(x, a) + w$ , we have  $\|y - y'\|^2 = \|\Theta - \Theta'\|_{\varphi(x, a)\varphi(x, a)^\top}^2$ , which shows that this assumption is indeed satisfied with the said choice of  $M$ .

Let us now prove that Assumption A2 holds: Let  $\Lambda$  be a random variable with probability distribution function

$$P(\lambda) \propto \exp\left(-\frac{1}{2}(\lambda - \hat{\Theta}_{t,(:,i)})^\top V_t (\lambda - \hat{\Theta}_{t,(:,i)})\right).$$

Notice that  $(\Lambda - \hat{\Theta}_{t,(:,i)})^\top V_t^{1/2} = Z \sim \mathcal{N}(0, I)$  has the standard normal distribution. Hence  $\mathbb{P}(\|Z_j\| > \alpha) \leq e^{-\alpha^2/2}$ . Thus, since  $\mathbb{P}(\|Z\| > \alpha) \leq m e^{-\alpha^2/(2m^2)}$ , we have

$$\begin{aligned} \mathbb{E} \left[ \left\| \left( \Theta_{*,(:,i)} - \hat{\Theta}_{t,(:,i)} \right)^\top V_t^{1/2} \right\|^2 \middle| \mathcal{F}_t \right] &= \mathbb{E} \left[ \|Z\|^2 \middle| \mathcal{F}_t \right] \\ &= \int_0^\infty \mathbb{P}(\|Z\|^2 > \epsilon) \leq 2m^3. \end{aligned}$$

Thus,

$$\begin{aligned} \mathbb{E} \left[ \left\| \left( \Theta_* - \hat{\Theta}_t \right)^\top V_t^{1/2} \right\|^2 \middle| \mathcal{F}_t \right] &\leq \mathbb{E} \left[ \left\| \left( \Theta_* - \hat{\Theta}_t \right)^\top V_t^{1/2} \right\|_F^2 \middle| \mathcal{F}_t \right] \\ &= \sum_{i=1}^n \mathbb{E} \left[ \left\| \left( \Theta_{*,(:,i)} - \hat{\Theta}_{t,(:,i)} \right)^\top V_t^{1/2} \right\|^2 \middle| \mathcal{F}_t \right] \\ &\leq 2nm^3. \end{aligned}$$

This shows that Assumption A2 is satisfied, thus finishing the proof.  $\square$

An immediate corollary of this is that Lazy PSRL will enjoy low regret when applied to linearly parametrized problems with Gaussian noise. We assume an effective stabilizing controller is available. This is necessary, as the noise may make the state arbitrarily large.

**Corollary 7.** *Consider Stabilized Lazy PSRL applied to a linearly parametrized problem with Gaussian noise with  $M$  as in Proposition 6. Let the underlying MDP satisfy Assumption A3. Suppose in time step  $t$ , the action chosen is  $\sigma_t$ -suboptimal. Then, for any time  $T$ , the regret of Stabilized Lazy PSRL satisfies  $R_T = \tilde{O}(\sqrt{T}) + \Sigma_T$ .*

*Proof.* The claim follows immediately from Proposition 6 and Theorem 3.  $\square$

## 7 EXPERIMENTS

In this section we illustrate the behavior of LAZY PSRL on a queuing and a web server control application.

### 7.1 Queuing Control Application

The queuing problem is described in (de Farias and Van Roy, 2003). The queue has a buffer size of 99. For time  $t$ , let  $x_t \in \{0, 1, \dots, 99\}$  be the state. The action  $a_t$  is the departure probability or service rate and is chosen from the set  $\{0.1625, 0.325, 0.4875, 0.65\}$ . Let  $p$  be the (unknown) arrival rate. The dynamics is defined as follows

$$x_{t+1} = \begin{cases} x_t - 1 & \text{with probability } a_t; \\ x_t + 1 & \text{with probability } p; \\ x_t & \text{otherwise.} \end{cases}$$

From state  $x_t = 0$ , transitions to states 1 and 0 happen with probabilities  $p$  and  $1 - p$ . From state  $x_t = 99$ , transitions to states 98 and 99 happen with probabilities  $a_t$  and  $1 - a_t$ . The loss function is  $\ell(x_t, a_t) = x_t^2 + 500p^2$ .

#### 7.1.1 Numerical Results

The purpose of this experiment is to show how the LAZY PSRL algorithm can take advantage of the problem structure to obtain better performance. We compare the LAZY PSRL algorithm with UCRL (Jaksch et al., 2010). For the LAZY PSRL algorithm, we use the Beta distribution  $\text{Beta}(1, 1)$  as the prior for the unknown parameter  $p$  (the conditions of our theorem can be seen to be satisfied along the lines of the previous section with  $M(x, a) = \text{const}$ ). The constant  $G$  in Figure 1 is chosen to be  $G = 2$ . The UCRL algorithm is an optimistic algorithm that maintains a confidence interval around each transition probability

$P(x'|x, a)$  and, in each round, finds the transition dynamics and the corresponding policy that attains the smallest average loss. Specifically, the algorithm solves the optimization problem  $\hat{P} = \operatorname{argmin}_P J(P)$ , where  $J(P)$  is the average loss of the optimal policy when the system dynamics is  $P$ . Then, the algorithm plays the optimal controller given the parameter  $\hat{P}$ . As we show next, the LAZY PSRL algorithm achieves lower average cost.

The time horizon in these experiments is  $T = 1,000$ . We repeat each experiment 10 times and report the mean and the standard deviation of the observations. Figure 3 shows average cost vs. number of rounds. Details of the implementation of the UCRL algorithm are in (Jaksch et al., 2010).

Figure 3 show the average cost of the algorithms. The LAZY PSRL algorithm outperforms the UCRL algorithm. We explain this observation by noting that the UCRL algorithm is learning components of the transition dynamics independently (400 components in total), while the LAZY PSRL algorithm takes advantage of the problem structure to speed up the learning.

## 7.2 Web Server Control Application

In this section we illustrate the behavior of LAZY PSRL on a simple LQR control problem. We choose an LQR control problem because it is a continuous state-action problem. Equally important is that this allowed us to compare the performance of LAZY PSRL to a competing method, the OFULQ algorithm of Abbasi-Yadkori (2012). The experiments go beyond the scope of the theory, as we did not use a stabilizing controller, though the control problem itself is such that the zero-dynamics (i.e., the dynamics under zero control) is stable, making it less likely that a stabilizing controller would be necessary for the method to work. In the next section we describe the control problem, which will be followed by the description of our results.

The problem is taken from Section 7.8.1 of the book by Hellerstein et al. (2004) (this example is also used in Section 3.4 of the book by Aström and Murray (2008)). An Apache HTTP web server processes the incoming connections that arrive on a queue. Each connection is assigned to an available process. A process drops the connection if no requests have been received in the last KEEPALIVE seconds. At any given time, there are at most MAXCLIENTS active processes. The values of the KEEPALIVE and MAXCLIENTS parameters, denoted by  $a_{ka}$  and  $a_{mc}$  respectively, are chosen by a control algorithm. Increasing  $a_{mc}$  and  $a_{ka}$  results in faster and longer services to the connections, but also increases the CPU and memory usage of the server. The state of the server is determined by the average processor load  $x_{cpu}$  and the relative memory usage  $x_{mem}$ . An *operating point of interest* of the system is given by  $x_{cpu} = 0.58, a_{ka} = 11s, x_{mem} = 0.55, a_{mc} = 600$ . A

linear model around the operating point is assumed, resulting in a model of the form

$$\begin{pmatrix} x_{cpu}^\Delta(t+1) \\ x_{mem}^\Delta(t+1) \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{21} \end{pmatrix} \begin{pmatrix} x_{cpu}^\Delta(t) \\ x_{mem}^\Delta(t) \end{pmatrix} + \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{21} \end{pmatrix} \begin{pmatrix} a_{ka}^\Delta(t) \\ a_{mc}^\Delta(t) \end{pmatrix} + \begin{pmatrix} w_1(t+1) \\ w_2(t+1) \end{pmatrix},$$

where  $(w_1(t+1), w_2(t+1))_t$  is an i.i.d. sequence of Gaussian random variables, with a diagonal covariance matrix  $\mathbb{E}[w(t+1)^\top w(t+1)] = \sigma^2 I$ . Note that these state and action variables are in fact the deviations from the operating point. We test  $\sigma = 0.1$  and  $\sigma = 1.0$  in our experiments. The matrices  $A, B, Q, R$  are included in the appendix.

### 7.2.1 Numerical Results

We compare the LAZY PSRL algorithm with OFULQ (Abbasi-Yadkori, 2012). For the LAZY PSRL algorithm, we use the standard normal distribution as the prior. The OFULQ algorithm is an optimistic algorithm that maintains a confidence ellipsoid  $D$  around the unknown parameter and, in each round, finds the parameter and the corresponding policy that attains the smallest average loss. Specifically, the algorithm solves the optimization problem

$$(\tilde{A}, \tilde{B}) = \operatorname{argmin}_{(A,B) \in D} J(A, B), \quad (4)$$

where  $J(A, B)$  is the average loss of the optimal policy when the system dynamics is  $(A, B)$ . Then, the algorithm plays the optimal controller given the parameter  $(\tilde{A}, \tilde{B})$ . The objective function  $J$  is not convex and thus, solving the optimistic optimization can be very time consuming. As we show next, the LAZY PSRL algorithm can have lower regret while avoiding the high computational costs of the OFULQ algorithm.

The time horizon in these experiments is  $T = 1,000$ . We repeat each experiment 10 times and report the mean and the standard deviation of the observations. Figure 4 shows regret vs. computation time. The horizontal axis shows the amount of time (in seconds) that the algorithm spends to process  $T = 1,000$  rounds. We change the computation time by changing constant  $G$  in Figure 1, i.e. by changing how frequent an algorithm updates its policy.<sup>2</sup> Details of the implementation of the OFULQ algorithm are in (Abbasi-Yadkori, 2012).

The first two subfigures of Figure 4 show the regret of the algorithms when the standard deviation of the noise is  $\sigma = 0.1$ . The regret of the LAZY PSRL algorithm is slightly worse than what we get for the OFULQ algorithm in this case. The LAZY PSRL algorithm outperforms

<sup>2</sup>For example, in Figure 4-(d), the average number of policy changes are (33.4, 45.2, 88, 127.1). In Figure 4-(c) the average number of policy changes are (5.6, 14.3, 30.8, 73.2, 140.2, 163).

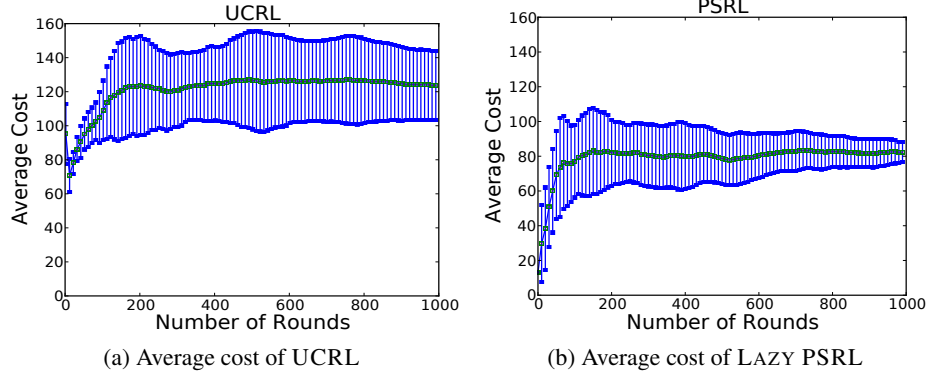


Figure 3: Average cost for a queuing problem.

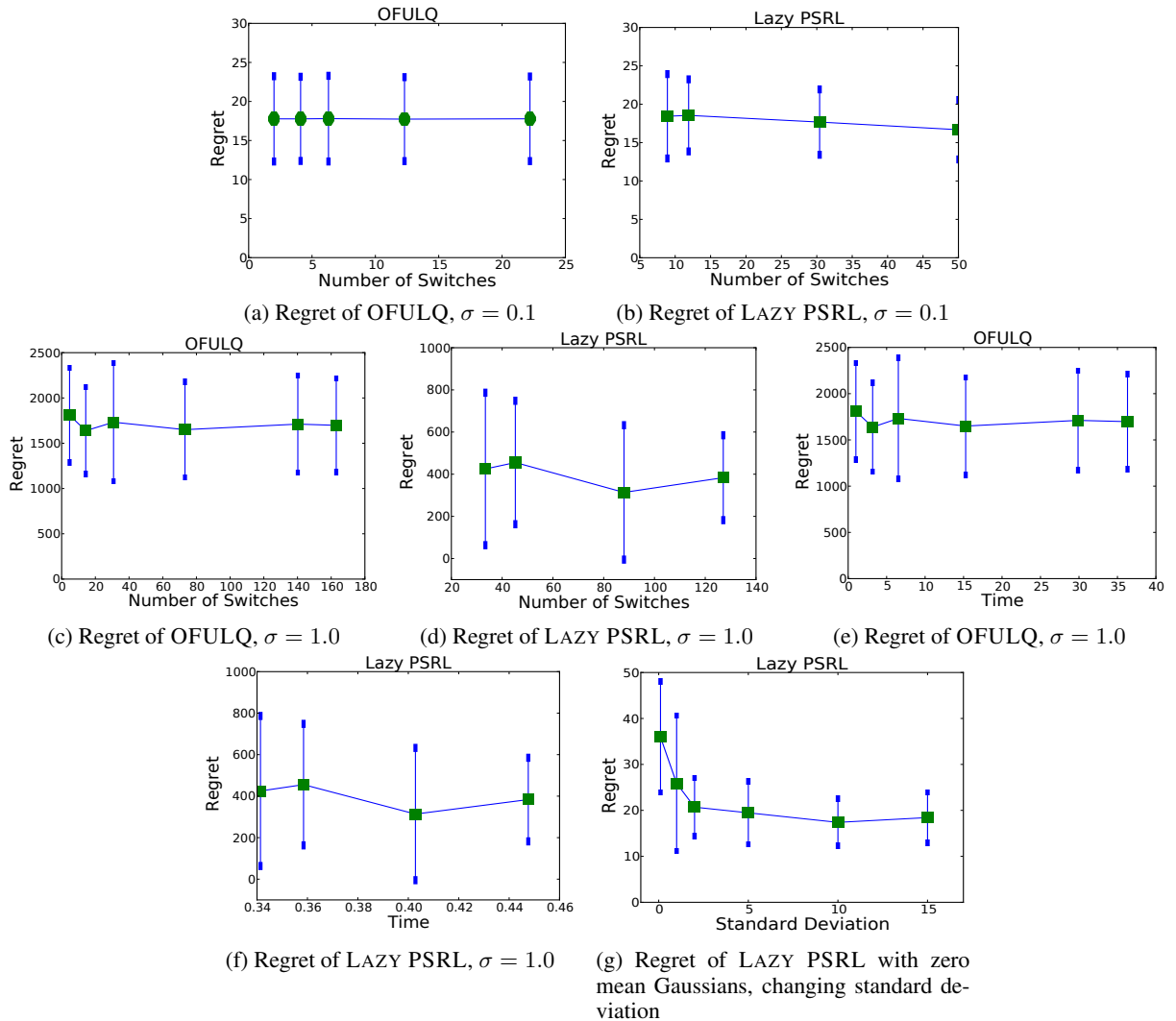


Figure 4: Regret for a web server control problem.

the OFULQ algorithm when the noise variance is larger (next two subfigures). We explain this observation by noting that a larger noise variance implies larger confidence ellipsoids, which results in more difficult optimistic opti-

mization problems (4). Finally, we performed experiments with different prior distributions. Figure 4-(e) shows regret of the LAZY PSRL algorithm when we change the prior.

## References

- Y. Abbasi-Yadkori. *Online Learning for Linearly Parametrized Control Problems*. PhD thesis, University of Alberta, 2012.
- Y. Abbasi-Yadkori and Cs. Szepesvári. Regret bounds for the adaptive control of linear quadratic systems. In *COLT*, 2011.
- A. Arapostathis, V.S. Borkar, E. Fernandez-Gaucherand, M.K. Ghosh, and S.I. Marcus. Discrete-time controlled Markov processes with average cost criterion: a survey. *SIAM Journal on Control and Optimization*, 31: 282–344, 1993.
- M. Araya-López, V. Thomas, and O. Buffet. Near-optimal BRL using optimistic local transitions. In *ICML*, 2012.
- J. Asmuth, L. Li, M. L. Littman, A. Nouri, and D. Wingate. A Bayesian sampling approach to exploration in reinforcement learning. In *UAI*, pages 19–26, 2009.
- Karl J. Aström and Richard M. Murray. *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton University Press, 2008.
- D. P. de Farias and B. Van Roy. Approximate linear programming for average-cost dynamic programming. In *NIPS*, 2003.
- A. Gopalan and S. Mannor. Thompson sampling for learning parameterized markov decision processes. In *COLT*, 2015.
- A. Guez, D. Silver, and P. Dayan. Scalable and efficient Bayes-adaptive reinforcement learning based on Monte-Carlo tree search. *Journal of Artificial Intelligence Research*, 48:841–883, 2013.
- A. Guez, D. Silver, and P. Dayan. Better optimism by Bayes: Adaptive planning with rich models. *CoRR*, abs/1402.1958, 2014.
- Joseph L. Hellerstein, Yixin Diao, Sujay Parekh, and Dawn M. Tilbury. *Feedback Control of Computing Systems*. John Wiley & Sons, Inc., 2004.
- A. Isidori. *Nonlinear Control Systems*. Springer Verlag, London, 3 edition, 1995.
- T. Jaksch, R. Ortner, and P. Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11:1563—1600, 2010.
- J. Z. Kolter and A. Y Ng. Near-Bayesian exploration in polynomial time. In *ICML*, 2009.
- I. Osband and B. Van Roy. Model-based reinforcement learning and the eluder dimension. In *NIPS*, 2014.
- I. Osband, D. Russo, and B. Van Roy. (More) efficient reinforcement learning via posterior sampling. In *NIPS*, 2013.
- M. Strens. A Bayesian framework for reinforcement learning. In *ICML*, 2000.
- W. R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25:285–294, 1933.
- N. Vlassis, M. Ghavamzadeh, S. Mannor, and P. Poupart. Bayesian reinforcement learning. In Marco Wieiring and Martijn van Otterlo, editors, *Reinforcement Learning: State-of-the-Art*, chapter 11, pages 359–386. Springer, 2012.

---

# Optimal expert elicitation to reduce interval uncertainty

---

**Nadia Ben Abdallah**  
Heudiasyc laboratory  
University of Technology of Compiègne  
Compiègne, France  
nadia.ben-abdallah@hds.utc.fr

**Sébastien Destercke**  
Heudiasyc laboratory  
University of Technology of Compiègne  
Compiègne, France  
sebastien.destercke@hds.utc.fr

## Abstract

Reducing uncertainty is an important problem in many applications such as risk and reliability analysis, system design, etc. In this paper, we study the problem of optimally querying experts to reduce interval uncertainty. Surprisingly, this problem has received little attention in the past, while similar issues in preference elicitation or social choice theory have witnessed a rising interest. We propose and discuss some solutions to determine optimal questions in a myopic way (one-at-a-time), and study the computational aspects of these solutions both in general and for some specific functions of practical interest. Finally, we illustrate the application of the approach in reliability analysis problems.

## 1 INTRODUCTION

When data on some quantity or model of interest is sparse or non-existing, elicitation, i.e., the process of extracting human judgement through questions, is often a valuable and sometimes the unique source of additional knowledge. There is a substantial literature dating back to the sixties on elicitation and is mainly related to probability encoding (Winkler, 1969; Spetzler and Stael von Holstein, 1975) and preference elicitation (Keeney et al., 1979). Elicitation is used in a broad range of fields including risk assessment (Cooke, 1991), reliability analysis, preference model elicitation (Viappiani and Kroer, 2013; Guerin et al., 2013), etc. to support assessment and decision making.

A critical part of the elicitation is then how to choose the questions to ask. Those need to be simple (i.e., do not require high cognitive effort) and in terms and format experts are familiar with. Furthermore, when the elicitation is conducted to reach some objective, for instance bringing an answer to a question, selecting the best alternative in a set, or estimating some quantity with a desired level precision, the process of information acquisition need to be optimal

for the elicitation to be effective and the least possible time or effort consuming.

How to choose sequences of optimal questions, or even the notion of optimal queries, has received surprisingly little attention when the aim is to reduce our uncertainty over some quantities. Indeed, the great majority of techniques to do so prescribe generic questions, without considering the consequences of answers on some final goal (Aspinall and Cooke, 2013) (the work of Curtis and Wood (Curtis and Wood, 2004), settled in a probabilistic context, is an exception). This contrasts with other fields such as preference elicitation of social choice theory, with works dating back two decades ago (Boutilier et al., 1997; Wang and Boutilier, 2003; Boutilier et al., 2006) and still thriving today (Viappiani and Kroer, 2013; Benabbou et al., 2014; Boutilier et al., 2013).

The goal of this paper is to explore similar ideas when the goal is to reduce interval uncertainty by asking successive simple questions to the experts. We want to develop querying strategies that are adaptive and optimal, i.e., that select at each stage of the elicitation the best questions based on the answers to the previous ones. In this paper, we focus on so-called myopic (Wang and Boutilier, 2003; Chajewska and Koller, 2000) strategies, where optimal questions are selected one-at-a-time.

The remainder of the paper is organized as follows. In Section 2, we formalize the sequential elicitation model for the problem of interval uncertainty reduction in the general case. Within this same section (Section 2.3), we describe different query selection strategies, and analyse their computational costs in the general case, which is an important aspect to consider in adaptive procedures. Section 3 then discusses the case of specific yet important (in practice) type of functions, namely monotonic and multi linear functions. In the last section, we illustrate how the approach can be used in reliability analysis.

## 2 GENERAL FRAMEWORK

### 2.1 PROBLEM STATEMENT

Let  $\Phi$  be a function mapping a set of  $n$  logically independent inputs  $(x_1, \dots, x_n)$ , each of them being defined on  $X_i$ , to an output  $y$  in  $Y$  :

$$\begin{aligned} \Phi : \mathbf{X} = \times_{i=1 \dots n} X_i &\rightarrow Y \\ \mathbf{x} = (x_1, \dots, x_n) &\mapsto \Phi(\mathbf{x}) = y. \end{aligned}$$

In this paper, we are interested in the situation where  $x_i$  is a precise but ill-known value, whose uncertainty is described by an interval  $X_i = [\underline{X}_i, \overline{X}_i] \subset \mathbb{R}$  of the real line. Such kind of uncertainty, where the true value is exact, is sometimes called epistemic (by opposition to aleatory). A natural way to quantify the amount of uncertainty in  $X_i$  is by its width

$$U_{X_i}(x_i) = U_{X_i} = \overline{X}_i - \underline{X}_i.$$

We also require the function  $\Phi$  to be continuous, so that the response  $y$  corresponding to the initial state of knowledge on the inputs lies in the bounded interval :

$$Y = \Phi(\mathbf{X}) = \left[ \min_{\mathbf{x} \in \mathbf{X}} \Phi(\mathbf{x}), \max_{\mathbf{x} \in \mathbf{X}} \Phi(\mathbf{x}) \right] = [\underline{Y}, \overline{Y}]. \quad (1)$$

**Example 1.** Consider the function  $\Phi(x_1, x_2, x_3) = x_1 x_2 - x_2 x_3$  with  $X_1 = X_2 = X_3 = [0, 1]$ , then we have

$$\underline{Y} = \Phi(\underline{X}_1, \overline{X}_2, \overline{X}_3) = -1; \overline{Y} = \Phi(\overline{X}_1, \overline{X}_2, \underline{X}_3) = 1.$$

The problem we are considering is the following : we want to reduce our uncertainty  $U_Y = \overline{Y} - \underline{Y}$  by asking question to experts, to attain some objectives. For instance, we may want to reduce the uncertainty under some threshold  $U_Y \leq s_0$  or simply reduce the most  $U_Y$  in a given number of questions. As expert elicitation is time-consuming and cognitively demanding for the expert, and economically expensive for the decision maker, we want to ask as few questions as possible, or to be the most effective possible on those questions we ask. In other words, we want the querying strategy to be optimal. This is what we develop in the next sections.

### 2.2 QUERIES AND ANSWERS

In expert elicitation in general, and when the elicitation is made of many successive questions, it is important to use simple questions that not require high cognitive effort (for understanding and answering) for the expert to be efficient throughout the interview. Possible simple queries formats include local bound queries (“ $x_i \leq \alpha$ ?”), pairwise comparison judgements (“ $x_i \leq x_j$ ?”), etc. (Braziunas and Boutilier, 2007).

In our method, we use questions of the type “ $x_i \leq \alpha$ ?”, with  $\alpha \in X_i$ . We denote such a query  $Q_i^\alpha$  and the set of possible queries  $\mathcal{Q} = \{Q_i^\alpha, i \in N = \{1, 2, \dots, n\}, \alpha \in X_i\}$ .

In the particular case of local bound queries, the set of possible answers  $\mathcal{A}$  is binary :  $\mathcal{A} = \{Yes, No\}$ . We recall that, for simplicity and for conciseness, we assume that the expert is an oracle, so the “I don’t know” answer is not considered here<sup>1</sup>. Note that the ideas presented in the paper could easily be applied to other sets of questions/answers  $\mathcal{Q}, \mathcal{A}$ , yet binary questions are the simplest and the most natural to ask to experts.

When a question  $Q_i^\alpha$  is asked and answer  $A \in \mathcal{A}$  is given,  $X_j$  remains unchanged for every  $j \neq i$ , while  $X_i$  is updated to  $X_i(Q_i^\alpha, A)$  as follows :

$$X_i(Q_i^\alpha, A) = \begin{cases} X_i \cap [-\infty, \alpha] & \text{if } A = Yes \\ X_i \cap [\alpha, -\infty] & \text{if } A = No \end{cases} \quad (2)$$

which satisfies  $X_i(Q_i^\alpha, A) \subseteq X_i$  and  $X_j(Q_i^\alpha, A) = X_j$  for every  $j \neq i$ . Consequently, the output uncertainty set is updated from  $Y$  into  $Y(Q_i^\alpha, A)$  :

$$Y(Q_i^\alpha, A) = \Phi(X_{-i} \times X_i(Q_i^\alpha, A)), \quad (3)$$

where  $X_{-i} = \times_{j \neq i} X_j$  denotes the Cartesian product of all unchanged intervals. As for any  $Q \in \mathcal{Q}$  and  $A \in \mathcal{A}$  we have  $Y(Q, A) \subseteq Y$  by simple interval inclusion, the following relation always holds :

$$U_Y \geq U_{Y(Q,A)} \quad (4)$$

therefore ensuring an uncertainty reduction.

**Example 2.** In Example 1, assume we ask the question  $Q_1^{0.5}$  and receive the answer *Yes*, then

$$X_1(Q_1^{0.5}, Yes) = [0, 0.5]$$

$$\overline{Y}(Q_1^{0.5}, Yes) = \Phi(\overline{X}_1, \overline{X}_2, \underline{X}_3) = 0.5.$$

### 2.3 QUERY SELECTION STRATEGIES

A query selection strategy corresponds to define and choose optimal questions. There are two main ways to do so : myopically, where questions are selected and asked one at a time, successively, and sequentially, where the set of successive questions is selected globally. Here, we retain the myopic approach for the following reasons : it is often simpler to solve, sometimes allowing for analytical exact solutions, and does not require to specify the number of asked questions in advance, a particularly interesting feature in iterative and interactive querying process.

1. Should the expert return “I don’t know” to  $Q_i^\alpha$ , then a simple strategy is to remove  $Q_i^\alpha$  (and possibly questions with similar values of  $\alpha$ ) from the question set  $\mathcal{Q}$  and then select the optimal one among the remaining ones.

The selection process of the myopic approach consists in solving the following optimization problem at each iteration :

$$Q^* = \arg \min_{Q \in \mathcal{Q}} U_Y(Q), \quad (5)$$

where  $U_Y(Q)$  is the uncertainty reduction induced by query  $Q$ . However, as the answer  $\mathcal{A}$  that will be given to  $Q$  is unknown, we face a typical problem of decision making under uncertainty.

In our case, the decision is a couple  $(i, \alpha) \in N \times X_i$ , the uncertain event is the answer to the question, and the outcome we want to maximize is the uncertainty reduction in the output  $Y$ . Re-writing the decision problem using notations of our query selection problem leads to the following characterization of the optimal queries :

$$Q^* = (i^*, \alpha^*) = \arg \min_{i \in N} \min_{\alpha \in X_i} U_Y(Q_i^\alpha), \quad (6)$$

which is a two stage optimization problem. First, we determine the optimal local bound value for each input  $i$ , and calculate the uncertainty reduction induced by that local query. Then, we select the entity  $i^*$  that leads to the highest uncertainty reduction in  $y$ .

---

**Algorithm : Iterative elicitation for uncertainty reduction**

---

```

Inputs :  $X_i (i \in N), s_0$ 
while  $U_Y \geq s_0$  do
  for  $i$  in  $N$  do
    Compute  $\alpha^* = \arg \min_{\alpha \in X_i} U_Y(Q_i^\alpha)$ 
    Compute  $U_Y(Q_i^{\alpha^*})$ 
  end for
   $i^* = \arg \min_{i \in N} U_Y(Q_i^{\alpha^*})$ 
  Ask query : “ $x_{i^*} \leq \alpha^*$ ?”
  Obtain answer
  Update  $X_{i^*}$ 
  Compute  $U_Y$ 
end while

```

---

In the following, we describe the computations involved in the first optimization step (the computation of  $U_Y(Q_i^\alpha)$ ) for a given  $i \in N$  for different decision criteria.

### 2.3.1 Maximin strategy

The maximin strategy corresponds to a pessimistic view, where the value  $U_Y(Q_i^\alpha)$  corresponds to the answer that yields the lowest uncertainty reduction :

$$\begin{cases} U_Y^{Mm}(Q_i^\alpha) = \max_{A \in \mathcal{A}} U_Y(Q_i^\alpha, A) \\ \alpha^{*,Mm} = \arg \min_{\alpha \in X_i} \max(U_Y(Q_i^\alpha, N_o), U_Y(Q_i^\alpha, Y_{es})) \end{cases}$$

We can show that solving the optimization problem to get the optimizing  $\alpha^*$  is equivalent to finding the intersection

of the two functions  $U_Y(Q_i^\alpha, Y_{es}), U_Y(Q_i^\alpha, N_o)$  of  $\alpha$ . The following propositions indicates that a general and efficient method to find the solution is to use a dichotomy search on the space  $[\underline{X}_i, \overline{X}_i]$

**Proposition 1.** *Functions  $U_Y(Q_i^\alpha, Y_{es})$  and  $U_Y(Q_i^\alpha, N_o)$  measuring the uncertainty level on  $Y$  induced by a positive and a negative answer to  $Q_i^\alpha$  and defined on  $X_i$  are – increasing and decreasing in  $\alpha$ , respectively, and – intersect at least once at  $M_i \subset X_i$  ( $M_i$  is a single point or an interval).*

**Proof.** *We have that*

$$U_Y(Q_i^\alpha, Y_{es}) = \max_{X_{-i} \times [\underline{X}_i, \alpha]} \Phi(\mathbf{x}) - \min_{X_{-i} \times [\underline{X}_i, \alpha]} \Phi(\mathbf{x}) \quad (7)$$

*To show that  $U_Y(Q_i^\alpha, Y_{es})$  is increasing in  $\alpha$ , we need to show that  $U_Y(Q_i^\alpha, Y_{es}) \leq U_Y(Q_i^\beta, Y_{es})$  for  $\alpha \leq \beta$ . This result follows from  $X_{-i} \times [\underline{X}_i, \alpha] \subseteq X_{-i} \times [\underline{X}_i, \beta]$ .*

*The same reasoning can be applied to*

$$U_Y(Q_i^\alpha, N_o) = \max_{X_{-i} \times [\alpha, \overline{X}_i]} \Phi(\mathbf{x}) - \min_{X_{-i} \times [\alpha, \overline{X}_i]} \Phi(\mathbf{x}) \quad (8)$$

*to show that  $U_Y(Q_i^\alpha, N_o)$  is decreasing in  $\alpha$ .*

*To demonstrate the second part of the proposition, simply observe that :*

$$\begin{aligned} \max_{\alpha \in X_i} U_Y(Q_i^\alpha, Y_{es}) &= U_Y(Q_i^{\overline{X}_i}, Y_{es}) = U_Y, \\ \max_{\alpha \in X_i} U_Y(Q_i^\alpha, N_o) &= U_Y(Q_i^{\underline{X}_i}, N_o) = U_Y, \end{aligned}$$

*where  $U_Y$  is the uncertainty before the question. As both functions have the same maximum, are continuous (since  $\Phi$  is), and are respectively increasing and decreasing in  $\alpha$ , they have at least one point of intersection.*

When  $M_i$  is an interval  $[\underline{M}_i, \overline{M}_i]$ , we simply take the middle point  $\alpha^{*,Mm} = \frac{\underline{M}_i + \overline{M}_i}{2}$ . In some situations, it may also happen that the intersection occurs on the bounds of  $X_i$  for all  $i$ , which means that the proposed optimal question is likely to be uninformative, unless the expert answer reduces the interval  $[\underline{X}_i, \overline{X}_i]$  to a point, which is unlikely. When such a scenario occurs, we use a different strategy that defines optimality as the highest reduction of uncertainty, no more on  $Y$ , but on  $X_i$ . This heuristic is equivalent, when  $X_i$ s are intervals, to choosing the largest interval  $i^* = \arg \max_i U_{X_i}$  and to pick the mid of this interval, i.e.,  $\alpha^{*,Mm} = \frac{\underline{X}_{i^*} + \overline{X}_{i^*}}{2}$ .

In Section 3, we will show that for specific functions, there are more efficient ways than a naive dichotomic search to determine  $\alpha^{*,Mm}$ .

### 2.3.2 Maximax strategy

While the maximin strategy is pessimistic, the maximax strategy is optimistic and takes as value  $U_Y(Q_i^\alpha)$  the answer that yields the highest uncertainty reduction :



$$\begin{cases} U_{Y(Q_i^\alpha)}^{MM} = \min_{A \in \mathcal{A}} U_{Y(Q_i^\alpha, A)} \\ \alpha^{*, MM} = \arg \min_{\alpha \in X_i} \min(U_{Y(Q_i^\alpha, No)}, U_{Y(Q_i^\alpha, Yes)}). \end{cases}$$

It is straightforward from Proposition 1 that the local bound optimization step leads to an optimal value  $\alpha^{*, MM}$  that coincides either with the upper or lower bound of  $X_i$ . The maximax strategy is therefore not interesting in our problem, as it will lead to questions that are most likely to receive a useless answer. We will therefore not retain this approach in this paper.

### 2.3.3 Hurwicz' strategy

Hurwicz's strategy evaluates the value of a question  $Q$  by a convex combination between the maximin and maximax strategies. It therefore allows to go from a pessimistic to an optimistic point of view and reads :

$$\begin{cases} U_{Y(Q_i^\alpha)}^{H(p)} = pU_{Y(Q_i^\alpha)}^{MM} + (1-p)U_{Y(Q_i^\alpha)}^{Mm} \\ \alpha^{*, H(p)} = \arg \min_{\alpha \in X_i} (p \min(U_{Y(Q_i^\alpha, Yes)}, U_{Y(Q_i^\alpha, No)}) + (1-p) \max(U_{Y(Q_i^\alpha, Yes)}, U_{Y(Q_i^\alpha, No)})). \end{cases}$$

Here,  $p \in [0, 1]$  is an optimism coefficient, and we retrieve the maximax and maximin strategies when  $p = 1$  and  $p = 0$ , respectively. Note that we can use the fact that for any  $\alpha \leq \underline{M}_i$  ( $\alpha \geq \overline{M}_i$ ), we have  $U_{Y(Q_i^\alpha, No)} \geq U_{Y(Q_i^\alpha, Yes)}$  ( $U_{Y(Q_i^\alpha, No)} \leq U_{Y(Q_i^\alpha, Yes)}$ ) to rewrite the above equations into :

$$\begin{cases} L = \min_{\alpha \in [\underline{X}_i, \inf M_i]} (pU_{Y(Q_i^\alpha, Yes)} + (1-p)U_{Y(Q_i^\alpha, No)}) \\ R = \min_{\alpha \in [\sup M_i, \overline{X}_i]} (pU_{Y(Q_i^\alpha, No)} + (1-p)U_{Y(Q_i^\alpha, Yes)}) \\ \alpha^{*, H(p)} = \min(L, R). \end{cases}$$

### 2.3.4 Bayesian strategy

Up to now, we have not considered any a priori information about the likelihood of answering Yes or No. However, this can lead to consider very unlikely answers, such as answering *Yes* to  $Q_i^{\underline{X}_i}$  (as is the case in the maximax strategy). One alternative is then the Bayesian strategy, where we assume the existence of a probability distribution  $P$  over the set of answers  $\mathcal{A}$ , this probability modelling our subjective beliefs about the likelihood of getting the different answers. We then evaluate a query  $Q$  by the expected reduction  $\mathbb{E}_P(U_{Y(Q, A)})$  of uncertainty on  $y$  induced by the possible answers :

$$\begin{cases} U_{Y(Q_i^\alpha)}^B = \mathbb{E}_P(U_{Y(Q_i^\alpha, A)}) = \sum_{A \in \mathcal{A}} P(A)U_{Y(Q_i^\alpha, A)} \\ \alpha^{*, B} = \arg \min_{\alpha \in X_i} (P(Yes|Q_i^\alpha)U_{Y(Q_i^\alpha, Yes)} + P(No|Q_i^\alpha)U_{Y(Q_i^\alpha, No)}). \end{cases}$$

When the available evidence suggests that a quantity  $x_i$  lies in an interval  $X_i$ , it is common to follow Laplace's indifference principle and quantify uncertainty by assuming a

uniform probability distribution over that set. Under this assumption, the probability of the positive and negative answers to a question  $Q_i^\alpha$  are proportional to the width of the sub-interval of  $X_i$  they lead to :

$$P(Yes|Q_i^\alpha) = P(\underline{X}_i \leq x_i \leq \alpha) = \frac{\alpha - \underline{X}_i}{\overline{X}_i - \underline{X}_i}$$

and

$$P(No|Q_i^\alpha) = P(\alpha \leq x_i \leq \overline{X}_i) = \frac{\overline{X}_i - \alpha}{\overline{X}_i - \underline{X}_i}.$$

These probabilities can then be modified according to the information we have (for instance, if we have reasons to think that the true value is closer to  $\overline{X}_i$ ).

**Remark 1.** Note that when  $U_{Y(Q_i^\alpha, Yes)} = U_{Y(Q_i^\alpha, No)} = U_{Y(Q_i^\alpha)}$ , then  $\mathbb{E}_P(U_{Y(Q_i^\alpha, A)}) = U_{Y(Q_i^\alpha)}$ , whatever the values of  $P$ . This means, among other things, that the function  $U_{Y(Q_i^\alpha)}^B$  has value  $U_{Y(Q_i^{\alpha^*, Mm})}^B = U_{Y(Q_i^{\alpha^*, Mm})}^{Mm}$ , since the minimax is obtained at the intersection  $M_i$  of  $U_{Y(Q_i^\alpha, Yes)}$  and  $U_{Y(Q_i^\alpha, No)}$ .

This means that we have :  $\min_{X_i} U_{Y(Q_i^\alpha)}^B = U_{Y(Q_i^{\alpha^*, B})}^B \leq U_{Y(Q_i^{\alpha^*, Mm})}^{Mm} = \min_{X_i} U_{Y(Q_i^\alpha)}^{Mm}$ , hence the expected uncertainty reduction with a Bayesian strategy is at least as high as the one obtained by the maximin strategy. However, in contrast with this latter, the Bayesian strategy does not offer guarantees about the uncertainty reduction, in the sense that the actual reduction may be lower than the expected one.

Also, while Proposition 1 means that  $\alpha^{*, Mm}$  can be obtained by a dichotomic search, this cannot be done in general for the Bayesian strategy, which therefore requires heavier computations.

**Example 3.** Consider the function  $\Phi(x_1, x_2, x_3) = x_1x_2 - x_2x_3 + x_2$  with  $X_1 = X_3 = [0.1, 1]$  and  $X_2 = [0, 1]$ . Figure 1 shows the various strategies for  $Q_2^\alpha$ . We can see that the maximin, the Laplacian and Hurwicz's strategies recommend respectively  $\alpha^{*, Mm} = 3/4$ ,  $\alpha^{*, B} = 1/2$ , and  $\alpha^{*, H(1/2)} = 0$ . Another remark is that  $U_{Y(Q_2^\alpha, Yes)}$  and  $U_{Y(Q_2^\alpha, No)}$  are both linear. We will see in the next section that this is true for multi linear functions in general.

## 3 QUERYING ON SPECIFIC FUNCTIONS

Here, we study what becomes of the previous strategies when applying them to specific functions. Indeed, finding an optimal strategy requires computing  $U_{Y(Q_i^\alpha, Yes)}$ ,  $U_{Y(Q_i^\alpha, No)}$  and their intersections, which comes down to finding bounds of  $\Phi$  over various domains (see Eqs. (7)-(8)). It is therefore important to identify those sub-cases for which computations can be simplified. More precisely, we look at monotonic functions and multi linear functions, that are both of practical interest.

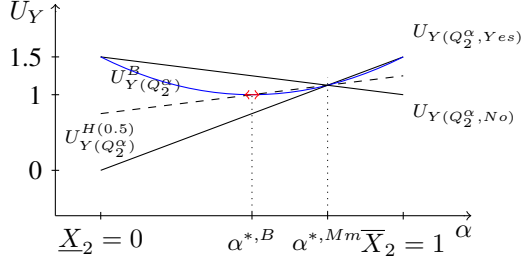


FIGURE 1 – Optimal recommendations of different query selection strategies.

### 3.1 MONOTONIC FUNCTIONS

Several application in diverse areas use monotonic functions, such as reliability analysis (Marichal, 2014), multi-criteria decision making (Grabisch and Labreuche, 2008), etc.

When considering such functions, either increasing or decreasing in each variable  $x_i$ , computations are greatly facilitated, as

$$U_Y = \Phi(\overline{X}_I, \underline{X}_{\overline{I}}) - \Phi(\underline{X}_I, \overline{X}_{\overline{I}}),$$

where  $I$  denotes the set of variables in which  $\Phi$  is increasing, and  $\overline{I}$  its complement.

Moreover, when  $\Phi$  is locally monotonic<sup>2</sup> with respect to each argument  $i$ , its upper and lower bounds are reached on the vertices of the hypercube  $\times_{i=1\dots n} X_i$ . Again, this may allow to reduce the computations involved in the calculation of  $U_Y$ .

### 3.2 MULTI LINEAR FUNCTIONS

A Multi-linear function over variables  $x_1, \dots, x_n$  is a polynomial form that can be written as

$$\Phi(x_1, \dots, x_n) = \sum_{A \subseteq N} d_A \prod_{i \in A} x_i \quad (9)$$

with  $d_A \in \mathbb{R}$  some real-valued coefficients. Such functions play an important role in many AI applications. As any pseudo-Boolean function can be rewritten in this form (Hammer and Rudeanu, 1968), they concern all problems where pseudo-Boolean functions have a role, such as cooperative game theory (Owen, 1972), multi-criteria decision-making (Grabisch and Labreuche, 2003), combinatorial optimization (Yannakakis, 1991), reliability theory (Bhattacharjya and Deleris, 2012; Marichal, 2014), etc. Multi linear functions also play an important role in

<sup>2</sup>  $\phi$  is locally monotone in  $x_i$  if, all other variables being fixed, it is either decreasing or increasing in  $x_i$ . Function  $\phi$  of Example 1 is locally monotone in  $x_2$ , as it is either increasing or decreasing in  $x_2$  once  $x_1$  and  $x_3$  are fixed.

inferences of Bayesian networks or related models (Darwiche, 2003; de Campos and Cozman, 2004).

From a computational point of view, having  $\Phi$  multi-linear presents different advantages. First, as  $\phi$  is locally monotonic in each variable (fixing every variable values but one in Eq. (9) gives a linear function, which is either increasing or decreasing), we know that its upper and lower bounds are reached on vertices of  $\times_{i=1\dots n} X_i$ . Second, provided  $0 \notin X_i$ , the maximin strategy will lead to a unique value  $\alpha^{*,M^m}$ , due to the fact that  $U_Y(Q_i^\alpha, Y_{es})$ ,  $U_Y(Q_i^\alpha, N_o)$  will be strictly increasing and decreasing, respectively (since bounds of Eq. (9) will be strictly monotone functions).

### 3.3 MULTI LINEAR MONOTONIC FUNCTIONS

Combining monotonicity and multi linear properties provide very interesting properties to compute our optimal strategies, and are still useful in several applications, such as reliability analysis that we use as a case study in the next section. The first property relates to the shape of  $U_Y(Q_i^\alpha, Y_{es})$  and  $U_Y(Q_i^\alpha, N_o)$

**Proposition 2.** *If  $\Phi$  is a multi linear function monotonic in each variable, then for every  $i \in N$ ,  $U_Y(Q_i^\alpha, Y_{es})$  and  $U_Y(Q_i^\alpha, N_o)$  are linear in  $\alpha$ .*

**Proof.** *If  $\Phi$  is monotonic in each variable, then in the first term of Eq. (7), the maximum is reached on the upper bounds of each  $X_i$ , i.e., on  $\overline{X}_j$  for all  $j \in N_{-i}$  and  $\overline{X}_i = \alpha$ , while the lower bound is reached on  $\underline{X}_j$  for all  $j \in N$  (independent of  $\alpha$ ). The function  $\Phi$  being linear in  $x_i$ ,  $\max \Phi(\mathbf{x})$  is therefore linear in  $\alpha$ , and so is  $U_Y(Q_i^\alpha, Y_{es})$ . The same reasoning applies to Eq. (8).*

This has several consequences on the computations of strategies :

- The maximin strategy recommends a unique query bound  $M_i$  in  $X_i$ , as  $U_Y(Q_i^\alpha, Y_{es})$  and  $U_Y(Q_i^\alpha, N_o)$  intersection will be a unique point ;
- Computing  $U_Y(Q_i^\alpha, Y_{es})$  and  $U_Y(Q_i^\alpha, N_o)$  will require only three computations, as they are linear (requiring each two evaluations) and as they have the same maximal value. Computing  $M_i$  then comes down to evaluate the intersection point of two lines ;
- Hurwicz’s solution will be reached either at the end-points of the interval  $X_i$  or will coincide with the maximin solution. The result follows from the fact that the convex combination of linear functions ( $U_Y(Q_i^\alpha, Y_{es})$  and  $U_Y(Q_i^\alpha, N_o)$ ) is also linear, and is therefore monotonic in  $\alpha$ .

Furthermore, we have the following property regarding the Bayesian strategy :

**Proposition 3.** *If  $\Phi$  is a multi linear function monotonic in each variable, the Bayesian strategy adopting a uniform distribution over  $X_i$  has a unique minimum  $\alpha$  in the interior of  $X_i$*

**Proof.** (sketch) Function  $U_{Y(Q_i^\alpha)}^B$  is convex since it is the sum of the product of two linear functions of  $\alpha$ , therefore it is quadratic and convex. In addition, it satisfies :  $U_{Y(Q_i^{X_i})}^B = U_{Y(Q_i^{\bar{X}_i})}^B$ .

Since it can not be a constant function (the scenario  $U_{Y(Q_i^\alpha, Y_{es})} = U_{Y(Q_i^\alpha, N_o)}$  for every  $\alpha \in X_i$  does not occur for multi linear functions), its global minimum exists, is unique, and is reached inside the interval  $X_i$ .

## 4 APPLICATION IN RELIABILITY ANALYSIS

When systems are complex or newly designed, full system dependability data are often too expensive and/or difficult to obtain, making it impossible to directly estimate quantities of interest. The common approach to improve the estimation of such quantities is to focus on enhancing the state of knowledge at the component-level, where information is more likely to be available either via measurements or expert elicitation.

In this section, we illustrate how our elicitation model can be used to refine the state of knowledge at the component level in order to estimate the reliability of a system. We begin by recalling some basic elements related to systems reliability and briefly describe the mathematical properties of the system function. Then, we describe and discuss the results of the proposed elicitation procedure on simple yet common system architectures, to finish by a real-world example involving railway safety systems.

### 4.1 PRELIMINARIES ON RELIABILITY ESTIMATION

Consider a network  $S$  with  $n$  components indexed in  $N = \{1, 2, \dots, n\}$ . We describe a static problem, i.e., we do not refer to time explicitly when describing the system behavior. Every component  $i$  is either operating or failing and its state is represented by a boolean variable  $e_i$  that associates 0 and 1 to the failed, working state, respectively. The system state is completely determined by the joint state of its components through the *structure function*  $\Phi_S$  – a boolean function. For the majority of systems, forming the class of semi-coherent systems, the structure function satisfies these three conditions :

- $\Phi_s$  non-decreasing in each  $e_i$
- $\Phi_s(0, 0, \dots, 0) = 0$
- $\Phi_s(1, 1, \dots, 1) = 1$ .

In reality, the state of component can not be determined exactly, and the usual framework is to assume that is a random variable. The probability that the component is functioning is called the elementary reliability :

$$p_i = Pr(e_i = 1).$$

When the components are independent, i.e., when their state variables are stochastically independent, the reliability of the system :

$$R = Pr(\Phi_s(e_1, \dots, e_n) = 1),$$

can be determined from the reliability of its components via the *reliability function*  $\Phi$  :

$$R = \Phi(p_1, \dots, p_n), \quad (10)$$

which is the multi linear extension of  $\Phi_s$  (Marichal, 2014) and so writes :

$$\Phi(p_1, \dots, p_n) = \sum_{A \subseteq N} d_A \prod_{i \in A} p_i$$

where coefficients  $d_A$  are the Mobius transform of the mass function associated with the structure function<sup>3</sup>. In practice, the exact expression of the reliability function can be directly generated using the inclusion-exclusion formula (Lin et al., 1976) based on determining the minimal path set (i.e., the minimal set of components that must be in working state that guarantees the functioning of the system) and cut sets (the minimal set of components such that if all of them fail, the system is guaranteed to fail whatever the value of the others components).

Therefore, when facing a new system with ill-known probabilities, we have a multi linear function  $\phi$  with interval-valued variables  $p_i$ , to which we can apply our previous findings.

### 4.2 CLASSICAL STRUCTURES

We first consider a bridge structure with 5 non redundant components. The reliability block diagram – a graphical depiction of the functional relationship between components – of this structure is given below :

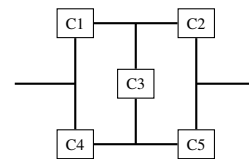


FIGURE 2 – Reliability block diagram of a series parallel system.

<sup>3</sup>. The Mobius transform of the mass function associated to  $\Phi_s$  is given by :

$$d_A = \sum_{B \subseteq A} (-1)^{|A|-|B|} \Phi_s(B).$$

The system reliability is given by :

$$\begin{aligned}
R &= p_1 p_2 + p_4 p_5 + p_1 p_3 p_5 + p_2 p_3 p_4 \\
&- p_1 p_2 p_3 p_4 - p_1 p_2 p_3 p_5 - p_1 p_2 p_4 p_5 - p_1 p_3 p_4 p_5 \\
&- p_2 p_3 p_4 p_5 + p_1 p_2 p_3 p_4 p_5.
\end{aligned}$$

We assume the initial state of knowledge to be the following :  $p_1 \in P_1 = [0.5, 0.92]$ ,  $p_2 \in P_2 = [0.2, 0.9]$ ,  $p_3 \in P_3 = [0.5, 0.9]$ ,  $p_4 \in P_4 = [0.4, 0.8]$ , and  $p_5 \in P_5 = [0.4, 0.85]$ . The system reliability ranges in the interval  $[0.3, 0.97]$ , so its initial uncertainty is 0.67.

We use the elicitation procedure to refine the state of knowledge over  $p_i$  ( $i \in \{1, \dots, 5\}$ ) via a sequence of queries on the elemental reliabilities. The objective is to reduce the system reliability uncertainty up to 0.05 (i.e.,  $s_0 = 0.05$ ), after which we stop asking questions.

To evaluate the efficiency of our procedure, we compare its performance to two basic strategies :

1. a random strategy that compares at each stage the reliability of component  $i$ , selected at random in  $N$ , with some random  $\alpha \in P_i$ . For the results to be significant, the performance of the strategy at each iteration is averaged over a high number (herein 1000) of runs.
2. a baseline strategy that asks at each stage about the most uncertain component, and the query bound is the midpoint of the largest interval (this strategy was referred to as the ‘‘halve largest Gap Strategy’’ in the context of preference elicitation (Boutillier et al., 2006)) :

$$Q_{Baseline}^* = \left( i^*, \frac{\bar{X}_{i^*} + \underline{X}_{i^*}}{2} \right)$$

where :

$$i^* = \arg \max_{i \in N} U_{X_i}.$$

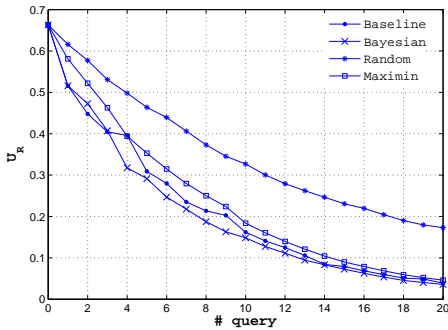


FIGURE 3 –  $U_R$  reduction using different selection strategies.

We implemented the elicitation procedure described in Section 2.3 assuming the true values to be the following :

$p_1^* = 0.6$ ,  $p_2^* = 0.7$ ,  $p_3^* = 0.65$ ,  $p_4^* = 0.7$ ,  $p_5^* = 0.78$ . Figure 3 shows the performance in terms of uncertainty reduction in  $R$  of our four strategies. The Bayesian slightly outperforms the baseline and the maximin strategies, but remains comparable to them, while all of them do much better than the random elicitation. In general, it takes twice the number of queries to the random strategy to reach the results of the other strategies (e.g., to divide uncertainty by half, it requires 10 questions for the random strategy, and about 5 for the others).

However, the performances of the Bayesian, maximin and baseline strategies highly depend on the initial situations. Figure 4 compares our previous experiment with another situation where the initial state of knowledge is very poor, i.e., a situation of near ignorance where  $P_i = [0.1, 0.9]$  for all  $i \in N$ . Results for both scenarios differentiated by the color and the line style (blue continuous lines and black dashed lines for the first and the second scenarios, respectively). The most notable difference between the two scenarios concerns the maximin strategy. Indeed, its performance in the second scenario significantly departs from the non-random strategies (to which it was very close in the first scenario). The maximin strategy is in this case probably too cautious, missing potentially good opportunities to reduce the uncertainty.

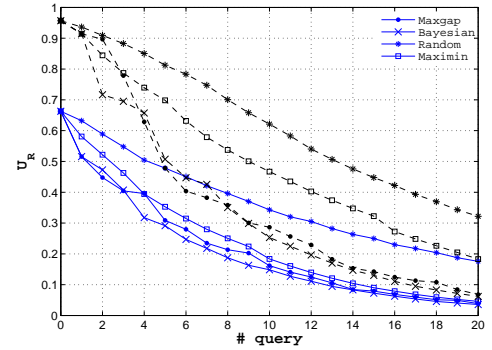


FIGURE 4 – Sensitivity of the performance of the selection strategies to the initial state of knowledge.

The good results of the baseline strategy for the bridge system are mainly due to the fact that every component is important in the system, hence gaining knowledge on any one of them reduces uncertainty in similar ways. This is not always true : consider a simple series parallel system composed of four independent and non-identical components (Fig. 5). The system reliability is :

$$R = p_1 p_2 p_3 p_4 + p_1 p_4 + p_2 p_4 + p_3 p_4 - p_1 p_3 p_4 - p_2 p_3 p_4 - p_1 p_2 p_4.$$

Let the initial state of knowledge on the elementary reliabilities be the following :  $p_1 \in [0.01, 0.99]$ ,  $p_2 \in [0.01, 0.99]$ ,  $p_3 \in [0.97, 0.99]$ ,  $p_4 \in [0.7, 0.9]$ , and the true values be :  $p_1 = 0.6$ ,  $p_2 = 0.7$ ,  $p_3 = 0.98$ ,  $p_4 = 0.8$ .

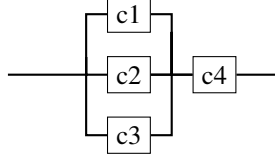


FIGURE 5 – Reliability block diagram of a series parallel system.

The results of the sequential elicitation procedure using the baseline strategy can be visualized in Figure 6 which plots the uncertainty on each component at every stage. A jump in the curve of component  $i$  at stage  $k + 1$  indicates that the  $k^{\text{th}}$  optimal query inquired about that component, and its magnitude corresponds to the uncertainty reduction after the question has been answered. Note that up to the 6<sup>th</sup> question, the strategy inquired about the reliability of components 1 and 2, being the most uncertain.

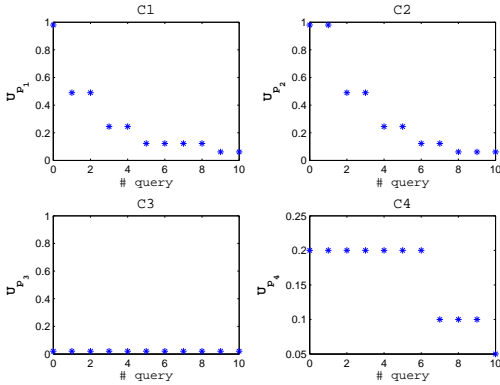


FIGURE 6 – Sequence of optimal components using the baseline strategy.

However, reducing uncertainty on components 1 and 2 does not reduce our global uncertainty, as shows Figure 7. In this case the baseline strategy performs actually very bad, not only compared to the maximin strategy, but also to the random up to the 6<sup>th</sup> query. This is due to the fact that the baseline strategy does not consider the importance components have on the overall system reliability.

### 4.3 REAL CASE SYSTEM

Up to now, we considered simple structures with distinct (non-redundant) components. However, the majority of real systems are complex and redundant, i.e., some of their components are duplicated. Redundancy ensures a backup in case of failure of one of the critical parts, and aims at increasing the overall reliability of the system.

When a system has redundancies, its reliability function is no longer multi linear, and depending on the redundancy architecture (parallel, triple modular, etc.), it becomes po-

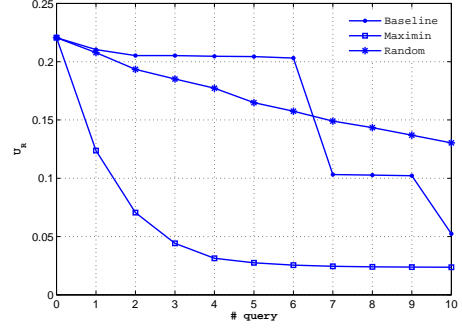


FIGURE 7 – Maximin, random, and baseline strategies for the case of series parallel system.

ynomial in the reliability of the redundant components, while remaining linear in the others. We are concerned here with the study of this type of systems/functions.

As a case study, we consider a real system used in the European railways traffic management system : the Radio Block Center system (RBC), whose role is to collect data about the position of trains and to provide movement authorisation (Flammini et al., 2006). Because of the relatively recent exploitation of the system, sufficient data to estimate the reliability of the RBC are lacking.

The RBC is composed of 5 different components, each of them being redundant. The architecture of the RBC is pictured in Figure 8, where the 2/3 symbol means that the subsystem composed of components 5 works if and only if at least 2 out of the three components work.

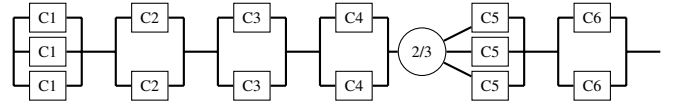


FIGURE 8 – Reliability block diagram of the RBC.

The reliability function can be written as :

$$R = (1 - (1 - p_1)^3)(1 - (1 - p_2)^2)(1 - (1 - p_3)^2)(1 - (1 - p_4)^2)p_{tmr};$$

with

$$p_{tmr} = (3p_5^2) - 2(p_5^3)(1 - (1 - p_6)^2).$$

We consider the case where some initial evidence suggests that the reliability of the RBC components ranges in  $[0.5, 1]$ , and that the true values are :  $p_1 = 0.83$ ,  $p_2 = 0.77$ ,  $p_3 = 0.8$ ,  $p_4 = 0.55$ ,  $p_5 = 0.72$ ,  $p_6 = 0.78$ . Results of the query strategy are plotted in Figure 9. Here, the maximin strategy outperforms the Bayesian one, which is consistent with Remark 1. The baseline does not do well and significant uncertainty reduction only occurs when asking about component 5, which is indeed the most important in this architecture.

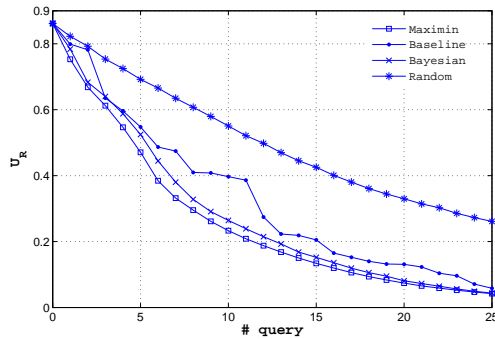


FIGURE 9 – Performance of the query strategies for the RBC system.

The computations involved in the elicitation procedure for this systems, and more complex systems in general, remain tractable as they only require optimization of polynomials and can still take advantage of the increasingness of function  $\Phi$ . This makes our procedure of practical use in real-time elicitation involving real experts – this will be the object of a forthcoming work concerned with the estimation of the RBC reliability using expert elicitation.

## 5 CONCLUSION

In this paper, we addressed the problem of optimal expert elicitation when the goal is to reduce interval uncertainty. We described different optimal querying strategies to determine the best question to ask at each stage of the procedure, studied their computational costs, and illustrated their use in a common estimation problem in reliability analysis.

For the particular problem of interval uncertainty reduction using local bound queries, the optimal elicitation approach proves to be effective and computationally tractable, especially for the maximin approach. We also discussed some cases, such as monotonic and multi linear functions, for which these computations are even easier. In future works, we plan to consider (1) more general uncertainty models, such as belief functions (Shafer, 1976) or probability sets (Augustin et al., 2014) which are particularly appealing to model, e.g. non-completely reliable experts (allowing for instance to relax the assumption that the expert is an oracle) and (2) other types of queries formats and answers, such as comparative assessments.

The strategies we described in this paper are myopic. Such strategies offer natural advantages (any-time stop, computational easiness), yet may select a sequence of questions that are globally sub-optimal, despite being locally optimal. A natural extension of this work is then to address the sequential approach for selecting the optimal set of queries to ask, and compare it with the myopic method. Clearly, this includes dealing with a computationally challenging pro-

blem, given the multistage nature of the optimization task, as well as some potential difficulties when choosing the values of strategies (e.g., the over-cautious nature of maximin could lead to strategies with very low average performances).

## Acknowledgments

This work was supported by the ANR (National Agency for Research) RECIF project (Reference : ANR-13-JS03-0007). It was carried out in the framework of the Labex MS2T.

## References

- Aspinall, W., Cooke, R., 2013. Quantifying scientific uncertainty from expert judgement elicitation. *Risk and Uncertainty Assessment for Natural Hazards*, 64.
- Augustin, T., Coolen, F., de Cooman, G., Troffaes, M., 2014. Introduction to imprecise probabilities. John Wiley & Sons.
- Benabbou, N., Perny, P., Viappiani, P., 2014. Incremental Elicitation of Choquet Capacities for Multicriteria Decision Making. In : *Proceedings of ECAI'14*. pp. 87–92.
- Bhattacharjya, D., Deleris, L. A., 2012. From reliability block diagrams to fault tree circuits. *Decision Analysis* 9 (2), 128–137.
- Boutilier, C., Brafman, R., Geib, C., Poole, D., 1997. A constraint-based approach to preference elicitation and decision making. In : *AAAI Spring Symposium on Qualitative Decision Theory*. Citeseer, pp. 19–28.
- Boutilier, C., Filmus, Y., Oren, J., 2013. Efficient vote elicitation under candidate uncertainty. *IJCAI*.
- Boutilier, C., Patrascu, R., Poupart, P., Schuurmans, D., 2006. Constraint-based optimization and utility elicitation using the minimax decision criterion. *Artificial Intelligence* 170 (8-9), 686–713.
- Braziunas, D., Boutilier, C., 2007. Minimax regret based elicitation of generalized additive utilities. In : *UAI*. pp. 25–32.
- Chajewska, U., Koller, D. and Parr, R., 2000. Making rational decisions using adaptive utility elicitation. In : *AAAI/IAAI*. pp. 363–369.
- Cooke, R., 1991. *Experts Uncertainty*. Oxford University Press.
- Curtis, A., Wood, R., 2004. Optimal elicitation of probabilistic information from experts. *Geological Society, London, Special Publications* 239 (1), 127–145.
- Darwiche, A., 2003. A differential approach to inference in bayesian networks. *Journal of the ACM (JACM)* 50 (3), 280–305.

- de Campos, C., Cozman, F., 2004. Inference in credal networks using multilinear programming. In : Proceedings of the Second Starting AI Researcher Symposium. pp. 50–61.
- Flammini, F., Marrone, S., Mazzocca, N., Vittorini, V., 2006. Modelling system reliability aspects of ertms/etcs by fault trees and bayesian networks. In : Proc. European Safety and Reliability Conference, ESREL. pp. 2675–2683.
- Grabisch, M., Labreuche, C., 2003. On the extension of pseudo-boolean functions for the aggregation of interacting criteria. *European Journal of Operational Research*.
- Grabisch, M., Labreuche, C., 2008. A decade of application of the choquet and sugeno integrals in multi-criteria decision aid. *4OR* 6 (1), 1–44.
- Guerin, J., Allen, T. E., Goldsmith, J., 2013. Learning cp-net preferences online from user queries. In : *Algorithmic Decision Theory*. Springer, pp. 208–220.
- Hammer, P., Rudeanu, S., 1968. *Boolean Methods in Operations Research and Related Areas*. Springer, Berlin Heidelberg, New York.
- Keeney, R. L., Raiffa, H., Rajala, D., 1979. Decisions with multiple objectives : Preferences and value trade-offs. *Systems, Man and Cybernetics, IEEE Transactions on* 9 (7), 403–403.
- Lin, P., Leon, B., Huang, T., 1976. A new algorithm for symbolic system reliability analysis. *Reliability, IEEE Transactions on R-25* (1), 2–15.
- Marichal, J.-L., 2014. Structure functions and minimal path sets. *arXiv preprint arXiv :1401.0803*.
- Owen, G., 1972. Multilinear extensions of games. *Management Sciences* 18, 64–79.
- Shafer, G., 1976. *A Mathematical Theory of Evidence*. Princeton University Press, Princeton, N. J.
- Spetzler, C., Stael von Holstein, C., 1975. Probability encoding in decision analysis. *Management Science*, 340–358.
- Viappiani, P., Kroer, C., 2013. Robots optimization of recommendation sets with the maximin utility criterion. *Proceedings of the Third Algorithmic Decision Theory International Conference*, 411–424.
- Wang, T., Boutilier, C., 2003. Incremental utility elicitation with the minimax regret decision criterion. In : *IJCAI*. pp. 309–318.
- Winkler, R., 1969. Scoring rules and the evaluation of probability assessors. *Journal of the American Statistical Association* 64 (327), 1073–1078.
- Yannakakis, M., 1991. Expressing combinatorial optimization problems by linear programs. *Journal of Computer and System Sciences* 43 (3), 441 – 466.

---

# Stochastic Integration via Error-Correcting Codes

---

**Dimitris Achlioptas**

Computer Science Department  
University of California, Santa Cruz  
Santa Cruz, CA 95064, USA

**Pei Jiang**

Computer Science Department  
University of California, Santa Cruz  
Santa Cruz, CA 95064, USA

## Abstract

We consider the task of summing a non-negative function  $f$  over a discrete set  $\Omega$ , e.g., to compute the partition function of a graphical model. Ermon et al. have shown that in a probabilistic approximate sense summation can be reduced to maximizing  $f$  over random subsets of  $\Omega$  defined by parity (XOR) constraints. Unfortunately, XORs with many variables are computationally intractable, while XORs with few variables have poor statistical performance. We introduce two ideas to address this problem, both motivated by the theory of error-correcting codes. The first is to maximize  $f$  over explicitly generated random affine subspaces of  $\Omega$ , which is equivalent to *unconstrained* maximization of  $f$  over an exponentially smaller domain. The second idea, closer in spirit to the original approach, is to use systems of linear equations defining Low Density Parity Check (LDPC) error-correcting codes. Even though the equations in such systems only contain  $O(1)$  variables each, their sets of solutions (codewords) have excellent statistical properties. By combining these ideas we achieve dramatic speedup over the original approach and levels of accuracy that were completely unattainable.

## 1 INTRODUCTION

The partition function of a graphical model with unnormalized probability function  $f$  over a domain  $\Omega$  is the integral (sum) of  $f$  over  $\Omega$ . The partition function is a central object of Bayesian statistics. While some inference tasks, such as MAP or MLE, can be completed without it, knowledge of (an approximation of) the partition function is necessary for marginalization, prediction, sampling, and model comparison, as a proper distribution is required. In general, partition function estimation is intractable [1] and, in practice, becomes problematic rapidly as  $|\Omega|$  increases.

To overcome this problem approximation schemes, such as MCMC [7], or variational methods [5] are commonly used. However, variational methods, in general, do not provide accuracy certificates/guarantees, while the mixing time of MCMC is similarly unpenetratable in most applications. In recent years, Ermon, Gomes, Sabharwal, and Selman have pioneered an alternative approach [14, 13, 15]. The general idea is to reduce the counting problem into a collection of random optimization problems, the final estimate being a statistic over the optima found. Each random problem amounts to maximizing  $f$  over a random set  $R \subseteq \Omega$ , defined via a random system of parity constraints. Realizing this idea, the WISH algorithm [14] is shown to yield a constant-factor approximation for the partition function *given access to an optimization oracle*. Importantly, the approximation guarantee requires the induced optimization problems to be solved to optimality, an assumption that clearly does not hold in general. Nevertheless, empirically, the WISH algorithm achieves remarkable accuracy compared to other established algorithms [14, 13, 15], leveraging the practical advancements in optimization software.

The idea of adding parity constraints originates in the work of Sipser [16] as a reduction technique. Most famously, it was used by Valiant and Vazirani [19] to reduce SAT to Unique SAT. A variant of the technique plays an important role in the proof of Toda's theorem [18]. The idea has since been applied to various counting problems including #SAT [11], # $k$ -SAT [17], and #CSP [12]. WISH [14] can be regarded as a natural generalization to weighted CSPs (or, equivalently, Markov Random Fields (MRFs)).

To provably approximate the partition function of an MRF with  $n$  variables, the parity constraints added must each have  $n/2$  variables on average. Unfortunately, the addition of such long constraints makes the MAP problem dramatically harder since each constraint (i) amounts to a clique involving half the variables of the MRF, and (ii) collapses the probability function whenever violated. In practice, this additional hardness can cause a MAP solver to submit dramatically suboptimal solutions under any reasonable time constraint, impairing the accuracy of estimation.



In all prior works the addition of random parity constraints is framed as *hashing* and the statistical properties of the resulting subsets of the domain is discussed in terms of independence properties of the corresponding families of hash functions. We break with this paradigm by taking a step back and asking: “how can we define subsets of the domain so that they are *computation-friendly* while having *good statistical properties*?” We answer the question twice, the two answers corresponding to two different notions of “friendliness” under the same notion of “goodness”.

Regarding goodness we will see that the key statistical property is pairwise *negative correlation* of membership, i.e., that for any two distinct  $\sigma, \sigma' \in \Omega$ , it should be that  $\Pr[\text{Both } \sigma, \sigma' \in R] \leq \Pr[\sigma \in R] \Pr[\sigma' \in R]$ . (Long parity constraints achieve this with equality.) Equivalently, conditioning on  $\sigma \in R$  should not make any  $\sigma' \neq \sigma$  more likely to be in  $R$  (but can make it less). Visualizing this as  $\sigma \in R$  exerting a repulsive force suggests an error-correcting code. Indeed, any linear error-correcting code  $C \subseteq \Omega = \{0, 1\}^n$  with  $2^{n-d}$  elements can be specified as  $C = \{\sigma \in \Omega : A\sigma = b\}$ , where  $A \in \{0, 1\}^{d \times n}$  is any rank  $d$  matrix and operations are over  $\text{GF}(2)$ , i.e., as the set of solutions to parity constraints.

Equipped with this idea, our first notion of computation-friendliness can be seen as “dimensionality reduction”. That is, instead of operating over  $\{0, 1\}^n$  and maximizing  $f$  over  $R$  by setting  $f(\sigma) = 0$  for  $\sigma \in \Omega \setminus R$ , we can operate over  $R$  directly by taking  $G \in \{0, 1\}^{n \times d}$  to be a *generator* of the subspace  $A\sigma = b$  and maximizing  $f(Gx + v)$  over  $x \in \{0, 1\}^d$ . We thus get an *unconstrained* optimization problem over a domain of size  $2^d$  instead of  $2^n$ . For any optimizer treating  $f$  as a black box, as is typical in MAP estimation, this makes optimizing  $f$  dramatically easier.

Our second notion of computation-friendliness can be seen as endowing  $\Omega \setminus R$  with a “gradient” (pointing towards  $R$ ), so that satisfying  $A\sigma = b$  does not impose significant computational burden. Again drawing insights from the theory of error-correcting codes, the idea is to desire the number of violated equations of  $A\sigma = b$  to be a function that has few local minima that are not global minima, i.e., not codewords, thus making its global minima easily accessible by some naive local method such as gradient descent. In other words, to make the optimizer’s life easy, we would like  $C = R$  to be an “easily decodable” code. This is precisely what we will achieve by taking the random sets  $R$  to correspond to the codewords of LDPC codes constructed by the Progressive-Edge-Growth construction [3].

Finally we note that independently of how the optimization problems are constructed, the *number* of instances that need to be solved can be reduced significantly in practice by using branch-and-bound. Combined with the two ideas mentioned above, this gives a dramatic speedup over WISH and entirely new levels of accuracy.

## 2 BACKGROUND

For the benefit of clarity, as in previous works, we will restrict our exposition to  $\Omega = \{0, 1\}^n$  and only approximate the partition function,  $Z$ , within a fixed constant factor, e.g., 32 (recall that, typically,  $Z \sim \exp(n)$ ). All ideas presented generalize readily to  $\Omega = D^n$  for any finite  $D$ .

### 2.1 BINARY MARKOV RANDOM FIELD

Given  $\Omega = \{0, 1\}^n$  and a collection of non-negative functions,  $\mathcal{F} = \{\psi_\alpha\}$ , defined on subcubes of  $\Omega$ , let

$$f(\sigma) = \prod_{\psi_\alpha \in \mathcal{F}} \psi_\alpha(\{\sigma\}_\alpha) ,$$

where  $\{\sigma\}_\alpha$  is the subset of variables entailed by  $\psi_\alpha$ . The *partition function* is the sum of  $f$  over all configurations:

$$Z = \sum_{\sigma \in \Omega} f(\sigma) .$$

### 2.2 ESTIMATION BY STRATIFICATION

We start by briskly revisiting (and, to some extent, reformulating) the groundbreaking work of Ermon et al. [14] connecting partition function estimation to optimization.

The first key idea is to stratify  $f$  over  $\Omega$  into quantiles and estimate  $Z$  by bounding from above and below the contribution of each quantile. Specifically, and w.l.o.g., assume that the configurations are sorted in descending order according to  $f$ , i.e.,  $f(\sigma_1) \geq f(\sigma_2) \geq \dots \geq f(\sigma_{2^n})$ . Let  $b_i = f(\sigma_{2^i})$ . Now, define the *lower sum* as

$$L := b_0 + \sum_{i=0}^{n-1} b_{i+1} 2^i$$

and the *upper sum* as

$$U := b_0 + \sum_{i=0}^{n-1} b_i 2^i .$$

Trivially,  $L \leq Z \leq U$ . Moreover,

$$\begin{aligned} 2L &= b_0 + \left( b_0 + \sum_{i=0}^{n-1} b_{i+1} 2^{i+1} \right) \\ &= b_0 + \sum_{i=0}^n b_i 2^i \geq U . \end{aligned}$$

Hence, if we compute  $b_0, b_1, \dots, b_n$ , taking any  $\hat{Z} \in [L, U]$  yields a 2-approximation of  $Z$ .

More generally, if for some integer  $c \geq 0$  and all  $i \in [n]$  an estimate  $\hat{b}_i \in [b_{i+c}, b_{i-c}]$  is available, then letting  $\hat{U}$  and  $\hat{L}$  be the counterparts of  $U$  and  $L$  with  $b_i$  replaced by  $\hat{b}_i$  we see that  $\hat{L} \leq L \leq Z \leq U \leq \hat{U}$  and  $\hat{U}/\hat{L} \leq 2^{2c+1}$ . Thus, any  $\hat{Z} \in [\hat{L}, \hat{U}]$  is a  $2^{2c+1}$ -approximation of  $Z$ , e.g., yielding a 32-approximation for  $c = 2$ .

### 2.3 STRATIFICATION BY THINNING

The second key idea is to estimate each  $b_i = f(\sigma_{2^i})$  as the maximum of  $f$  over a random set  $R_i \subseteq \Omega$  of (expected) size  $2^{n-i}$ . As mentioned, the essential requirement for this approach to work is *pairwise negative correlation* of membership in  $R_i$ . Including each element of  $\Omega$  in  $R_i$  independently with probability  $2^{-i}$  achieves this trivially but at the cost of an exponentially large, and thus inoperable, representation of  $R_i$ . The foundation of this entire line of research has been that it is possible to achieve *pairwise* independence for membership in  $R$  in compact form via hashing. We introduce a somewhat more general, and ultimately more fruitful, point of view reflected in our definition of Thinning Sets below.

**Thinning Sets.** A random variable  $R_i$  taking values in  $2^\Omega$  is an  $i$ -thinner if

- $\forall \sigma, \Pr[\sigma \in R_i] = 2^{-i}$  (Uniform)
- $\forall \sigma \neq \sigma', \Pr[\sigma \in R_i \wedge \sigma' \in R_i] \leq 2^{-2i}$  (Universal)

Given an  $i$ -thinner  $R_i$  and a solver capable of maximizing  $f$  over  $R_i$ , estimating  $b_i$  is entirely straightforward. Theorem 1 below is identical to the main result of [14], except for thinning sets replacing hash functions (for completeness we prove Theorem 1 in Section 5.)

**Theorem 1** ([14]). *Let  $R_i$  be any  $i$ -thinner random variable. Let  $\{m_j\}_{j=1}^t$  be i.i.d. random variables distributed as  $m_j = \max_{\sigma \in R_i} f(\sigma)$ . If  $M = \text{median}(m_1, \dots, m_t)$ , then for every  $c \geq 2$ ,*

$$\Pr[b_{i+c} \leq M \leq b_{i-c}] \geq 1 - 2 \exp\left(-\frac{t}{2}(1 - 2^{-c+1})^2\right).$$

One way to create an  $i$ -thinner is to take the solutions of a random system of linear equations over  $\text{GF}(2)$ , i.e., modulo 2. Let  $A \sim \text{Ber}(m \times n)$  denote that  $A$  is an  $m \times n$  random matrix whose entries are independent random variables with  $\Pr[a_{ij} = 1] = \Pr[a_{ij} = 0] = 1/2$ , for all  $i, j$ .

**Random Linear Code.** *The random set*

$$R_i = \{\sigma \in \{0, 1\}^n : A\sigma = b\} \quad (1)$$

*is an  $i$ -thinner if  $A \sim \text{Ber}(i \times n)$  and  $b \sim \text{Ber}(i \times 1)$ .*

In coding theory the set  $R_i$  in (1) is known as a random linear code, while the distribution  $A \sim \text{Ber}(i \times n)$  is known as the Shannon ensemble. Note that the rows of  $A$  have, on average,  $n/2$  non-zero entries. We will refer to it as the *dense parity* ensemble, to distinguish it from other distributions on  $\{0, 1\}^{i \times n}$  which we will encounter shortly. By Theorem 1, we can thus estimate  $b_i$  given an oracle  $\mathcal{O}$  for

$$\max_{\substack{\sigma \in \{0, 1\}^n \\ A\sigma = b}} f(\sigma). \quad (2)$$

The idea of adding long random parity constraints to achieve unweighted counting, e.g., to count the number of satisfying assignments of a CNF formula goes back to [11]. Ermon et al. in [14], after  $i$ -thinning  $\Omega$  in the manner above, solved the optimization problem (2) with ToulBar2 [2], dedicated software for MAP estimation in graphical models (the parity constraints added as factors to  $f$  evaluating to 0 when violated). In later work [13], the authors translated (2) to an Integer Linear Program, thus bringing to bear CPLEX, a powerful commercial optimization software. Finally, for reasons to be discussed shortly, in [15], the dense parity ensemble was replaced by the *sparse parity ensemble* wherein the entries of  $A$  are i.i.d. Bernoulli random variables where  $\Pr[a_{ij} = 1] = p < 1/2$ .

## 3 OUR CONTRIBUTION

### 3.1 RANDOM AFFINE MAPS

Instead of starting with  $\Omega = \{0, 1\}^n$  and restricting it via  $i$  random parity equations to a subset  $R_i$  of (expected) size  $2^{n-i}$ , we will *start* with  $\{0, 1\}^{n-i}$  and generate  $R_i$  as the image of  $\{0, 1\}^{n-i}$  under a random affine transformation  $g : \{0, 1\}^{n-i} \rightarrow \{0, 1\}^n$ , where  $g(x) = Ax + b$ . Thus, instead of solving the constrained optimization problem

$$\max_{\substack{\sigma \in \{0, 1\}^n \\ A\sigma = b}} f(\sigma),$$

we will solve the *unconstrained* optimization problem

$$\max_{x \in \{0, 1\}^{n-i}} (f \circ g)(x),$$

over the *exponentially* smaller set  $\{0, 1\}^{n-i}$ . The benefit of such dimensionality reduction increases with  $i$ , i.e., smaller  $R_i$ , in contrast to thinning by parity constraints which typically has worsening behavior as  $i$  is increased.

### 3.2 LOW DENSITY PARITY CHECK CODES

In certain settings, such as when performing “light” thinning or when the function  $f$  can be optimized better than black box, operating directly on the restriction of  $\Omega$  induced by parity constraints is preferable to operating through a random affine map. In these settings, instead of forming the constraint matrix  $A$  by having its entries be i.i.d. Bernoulli random variables (either sparse or dense) we will take  $A$  to be the parity check matrix of a Low Density Parity Check (LDPC) code. As we demonstrate experimentally, this has a stunning effect on the performance of CPLEX.

In the eyes of a solver operating on the variable representation of  $\Omega$  (as opposed to a local search solver) a 3-XOR is greatly preferable to an  $(n/2)$ -XOR, even though both shrink the domain by half. This is because repairing a violated constraint of arity  $k$  represents a  $k$ -way choice, i.e.,

a branching factor of  $k$ . This motivated the introduction of sparse i.i.d. Bernoulli parity check matrices in [11] and later in [15]. While a step in the right direction, this does not go far enough. To cover the remaining distance, we exploit insights from the modern theory of LDPC codes.

Imagine a code  $\mathcal{C} = \{\sigma \in \{0, 1\}^n : A\sigma = b\}$ , for some fixed matrix  $A$  and vector  $b$ . Imagine further that  $\sigma \in \mathcal{C}$  is transmitted along a channel that *erases* a subset of the bits of  $\sigma$ , so that the recipient receives  $\tau \in \{0, 1, *\}^n$ . Clearly, equations (checks) with no  $*$  variables offer no new information regarding  $\sigma$ . On the other hand, equations with two or more  $*$  variables are ambiguous, as they can be satisfied in multiple ways. But any equation with exactly one  $*$  variable is ideal: its erased bit can be recovered unambiguously; moreover, this recovery may cause other equations that had two  $*$  variables to now only have one. Such a cascade of “safe steps” will recover  $\sigma$  unless it encounters a *stopping set*: a non-empty set  $V$  of erased bits, such that no equation entails exactly 1 element of  $V$ . The amazing performance of LDPC codes is, to first order, due to the absence of small stopping sets. Thus, if  $\tau$  does not have too many erased bits, safe steps will suffice to recover  $\sigma$ .

To readers familiar with satisfiability algorithms the parallel between “safe” decoding and unit-clause propagation (UCP) will be immediate. The linear equations defining code  $\mathcal{C}$  can be thought of as a formula  $F$  very carefully designed to have the following property: if one selects a random subset of variables and assigns them random values (subject only to no empty clause being created), then for the vast majority of random choices (corresponding to the unerased bits in the communication setting) the residual formula should be solvable by UCP alone. It is not hard to imagine that adding such a formula  $F$  to a formula  $F'$  will induce little “additional hardness” to any solver capable of recognizing the presence of “safe” choices: as soon as enough variables are instantiated to get within the “radius of attraction” of a solution to  $F$ , the solver devolves to a “safe choice decoder”, setting variables at a rapid pace with minimal branching. We conjecture that this is precisely what causes the stunning improvement we observe in the performance of CPLEX when switching from dense/sparse parity ensembles to LDPC codes. Unfortunately, verifying this directly is non-trivial as CPLEX is commercial software. As implicit evidence we offer the observed complete insensitivity of stochastic local search to the structure of  $A$  (we use LocalSolver [9] in our experiments).

### 3.3 BRANCH AND BOUND

Recall that to form our estimate  $\widehat{Z}$  we multiply each  $\widehat{b}_i = f(\sigma_{2^i})$  by  $2^i$ . As a result, in most cases,  $\widehat{Z}$  is dominated by the contribution of a set of quantiles  $I \subseteq [n]$ , where  $|I| \ll n$ . (Indeed, in physical terms, failure of this to be true is the signature of criticality.) With this observation

in mind, rather than estimating all  $\{\widehat{b}_i\}_{i=1}^n$  in sequence, we can save computation by starting with  $S = \{\widehat{b}_0, \widehat{b}_n\}$  and estimating more and more quantiles until sufficient accuracy is achieved. In particular, simply enlarging  $S$  by the unestimated quantile of greatest remaining potential contribution, gives a speedup ranging from 2x to 10x in our experiments, with 7x being the most common case.

## 4 RANDOM AFFINE MAPS

Throughout this section let  $d := n - i$ , where  $i \in [n]$ . Let  $A \in \{0, 1\}^{n \times d}$  be a matrix of rank  $d$  and let  $b \in \{0, 1\}^n$ . If  $g(x) = Ax + b$ , then the image of  $\{0, 1\}^d$  under  $g$  is an affine subspace of  $\{0, 1\}^n$  containing  $2^d$  distinct elements (since  $A$  has full rank). Let  $\{0, 1\}_d^{n \times d}$  denote the set of all full rank, i.e., rank  $d$ , matrices in  $\{0, 1\}^{n \times d}$ .

**Theorem 2.** *Let  $A$  be uniform over  $\{0, 1\}_d^{n \times d}$  and let  $v$  be uniform over  $\{0, 1\}^n$ . The image of  $\{0, 1\}^d$  under  $x \mapsto Ax + v$  is an  $(n - d)$ -thinner.*

Theorem 1 immediately implies the following.

**Corollary 1.** *Let  $A$  be uniform over  $\{0, 1\}_d^{n \times d}$  and let  $v$  be uniform over  $\{0, 1\}^n$ . Let  $\{m_j\}_{j=1}^t$  be i.i.d. random variables distributed as*

$$\max_{x \in \{0, 1\}^d} f(Ax + v) .$$

*If  $M = \text{median}(m_1, \dots, m_t)$ , then for every  $c \geq 2$ ,*

$$\Pr[b_{i+c} \leq M \leq b_{i-c}] \geq 1 - 2 \exp\left(-\frac{t}{2}(1 - 2^{-c+1})^2\right) .$$

In other words, replacing long parity constraints with random affine maps, retains all statistical guarantees while giving rise to optimization instances over  $\{0, 1\}^{n-i}$  instead of  $\{0, 1\}^n$ . As in the estimation of the partition function,  $i$  ranges from 1 to  $n$ , at some point it becomes much more efficient to operate on  $f \circ g$  in the reduced domain than to operate on  $f$  on  $\Omega$ . Moreover, because  $|R_i| = 2^{n-i}$  deterministically, rather than in expectation, the variance of each estimate  $m_i$  is smaller than for a random linear code.

To sample uniformly from  $\{0, 1\}_d^{n \times d}$  it is convenient and efficient to employ rejection sampling: trivially generate  $A \sim \text{Ber}(n \times d)$  and accept only if  $\text{rank}(A) = d$ . Uniformity follows from the uniformity of  $A \sim \text{Ber}(n \times d)$  over  $\{0, 1\}^{n \times d}$ . By Lemma 1, the number of trials needed is a geometric random variable with mean less than 4.

**Lemma 1.**  $|\{0, 1\}_d^{n \times d}| > 2^{dn-2}$ .

### 4.1 EVALUATION

In Figure 1 we compare parity constraints vs. affine maps on the  $10 \times 10$  Ising grid with  $F = 0.1$  and  $C = 1.0$ , a noted hard problem in [14]. For the exact definition of

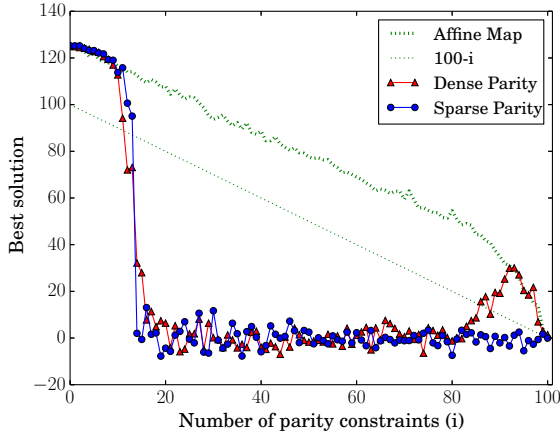


Figure 1:  $10 \times 10$  Ising grid with  $C = 1.0$  and  $F = 0.1$ . *Experiment:* for  $i \in [0, 100]$  generate  $R_i$  via  $i$  random parity constraints, or as a random affine subspace of dimension  $n - i$ ; seek  $\max_{\sigma \in R_i} f(\sigma)$  for 30 seconds (see legend). *Plot:* for each  $i$ , repeat the experiment 10 times and report the binary logarithm of the median value found. We also plot the number  $100 - i$  as a visual aid.

$f$  see (9) in Section 7. The parity constraints are generated from the dense parity ensemble and the sparse parity ensemble adopted in [15] and we use the ILP formulation proposed in [13], with CPLEX being the solver. To optimize  $f \circ g$  for random affine maps we use LocalSolver [9], since CPLEX is an ILP solver and does not natively support affine transformations over  $\text{GF}(2)$ . While not as strong as CPLEX on constrained optimization problems, LocalSolver is specialized in stochastic local search under black-box evaluation, hence a suitable choice for our setting.

As can be seen in Figure 1, when there are more than, roughly, 10 parity constraints, the performance of both random parity ensembles deteriorates rapidly, in sharp contrast to the robust performance of LocalSolver under affine maps. Note that since the  $y$ -axis is in  $\log_2$ -scale and each  $b_i$  contributes roughly  $b_i 2^i$  to the partition function, the fact that the solutions found by LocalSolver are nearly parallel to the line  $100 - i$  imply that the under-performance of optimization under parity constraints is highly relevant and will have dramatic effect on the accuracy of estimation.

Moreover, as shown in Figure 2, the best solutions found under parity constraints in 10 minutes are still inferior to those found via random affine maps in 30 seconds. In particular, when there are 50 constraints, CPLEX cannot find a solution better than the initial one under either parity ensemble, suggesting that the hardness of optimization under parity constraints overwhelms the solver. Notably, the original MAP inference  $\max_{\sigma \in \Omega} f(\sigma)$  can be solved to provable optimality in 0.1 second by CPLEX, highlighting that the hardness is due to the parity constraints.

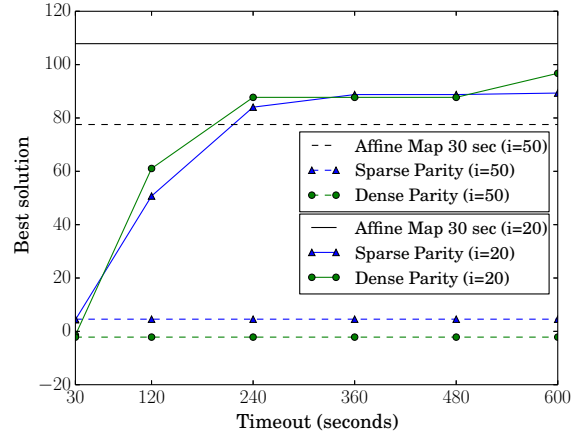


Figure 2:  $10 \times 10$  Ising grid with  $C = 1.0$  and  $F = 0.1$ . *Experiment:* for  $i \in \{20, 50\}$  generate  $R_i$  via  $i$  random parity constraints; seek  $\max_{\sigma \in R_i} f(\sigma)$  with a timeout of  $t \in \{30, 120, 240, 360, 480, 600\}$  seconds (see legend). *Plot:* for each pair  $(i, t)$  repeat the experiment 10 times and report the binary logarithm of the values found.

## 4.2 PROOF OF THEOREM 2 AND LEMMA 1

We will refer to  $A$  as a *generator* matrix for the subspace, which we will represent by the pair  $(A, v)$ .

**Lemma 2.** *If  $A$  is uniform over  $\{0, 1\}_d^{n \times d}$  and  $v$  is uniform over  $\{0, 1\}^n$ , then  $(A, v)$  is uniform over all affine subspaces of dimension  $d$ . In particular, for any fixed  $v_0$ , the subspace  $(A, v_0)$  is uniform over all affine subspaces of dimension  $d$  that contain  $v_0$ .*

*Proof.* It suffices to prove the second statement as the independence of  $A$  and  $v$  implies the first.

We will prove that for any fixed  $v_0$  and any affine subspace  $\mathcal{A}$  of dimension  $d$  that contains  $v_0$ , the number of matrices  $A \in \{0, 1\}_d^{n \times d}$  such that  $(A, v_0) = \mathcal{A}$  is independent of  $\mathcal{A}$ .

Clearly,  $(A, v_0) = \mathcal{A}$  iff the columns of  $A$  are linearly independent elements of  $\mathcal{A}$ . For  $k \in [d]$ , let  $a_k$  be the  $k$ -th column of  $A$  and let  $a_0 = \mathbf{0}$ . Linear independence is equivalent to  $a_k \notin \text{span}(a_0, \dots, a_{k-1})$  for all  $k \in [d]$ . Since  $|\text{span}(a_1, \dots, a_k)| = 2^k$  if the first  $k$  columns are linearly independent, we see that for every  $k \in [d]$  there are  $2^d - 2^{k-1}$  valid choices for the  $k$ -th column.  $\square$

*Proof of Theorem 2.* By the first part of Lemma 2,  $(A, v)$  is uniform over all affine subspaces of dimension  $d$ . Therefore, by symmetry,  $\Pr[\sigma \in (A, v)]$  is the same for all  $\sigma \in \{0, 1\}^n$ . Since  $(A, v)$  contains  $2^d$  elements, uniformity follows. To prove universality we need to show that  $\Pr[\sigma' \in (A, v) \mid \sigma \in (A, v)] \leq \Pr[\sigma' \in (A, v)]$ . For this we first observe that, by the second part of Lemma 2,

$$\Pr[\sigma' \in (A, v) \mid \sigma \in (A, v)] = \Pr[\sigma' \in (A, \sigma)] .$$

Since  $\sigma \neq \sigma'$ , this last probability equals the probability that  $\tau = \sigma' - \sigma \neq \mathbf{0}$  belongs in  $(A, 0)$ . Let  $a_k$  be the  $k$ -th column of  $A$  and let  $A_k$  comprise the first  $k$  columns of  $A$ . If we generate  $A$  column by column we see that this last probability equals  $1 - \prod_{k=1}^d \Pr[a_k \notin \text{span}(A_{k-1} \cup \{\tau\})]$  which is the same for all  $\tau \neq \mathbf{0}$ .  $\square$

*Proof of Lemma 1.* If we construct  $A$  column by column then, as shown in Lemma 2, there are  $2^n - 2^{k-1}$  choices for the  $k$ -th column that lead to  $A$  being full rank. Induction thus shows  $\prod_{k=1}^d (2^n - 2^{k-1}) \geq \frac{1}{4} (2^{dn} + 2)$ .  $\square$

## 5 THINNING AS ERROR-CORRECTION

Let us start by deriving the statistical desiderata of thinning sets from first principles. This will illuminate the suitability of error-correcting codes for thinning and offer insight. Recall that our goal is to estimate  $b_i = f(\sigma_{2^i})$ , for  $i \in [n]$ . We start by observing that for this it suffices to construct a random variable  $m_i$  such that for some (small) integer  $c$  and any  $\epsilon > 0$ ,

$$\Pr[m_i \leq b_{i-c}] \geq 1/2 + \epsilon \quad (3)$$

$$\Pr[m_i \geq b_{i+c}] \geq 1/2 + \epsilon \quad (4)$$

This is because if we take  $\hat{b}_i$  to be the median of  $t$  independent realizations of  $m_i$ , by Hoeffding's inequality,

$$\Pr[b_{i+c} \leq \hat{b}_i \leq b_{i-c}] \geq 1 - 2 \exp(-2\epsilon^2 t) \ .$$

Thus, in order for  $\Pr[m_i \in [b_{i+c}, b_{i-c}]] = 1 - \exp(-\Theta(s))$  it suffices to take  $O(s/\epsilon^2)$  samples.

Achieving (3) is trivial. Let  $\Omega_j = \{\sigma_1, \sigma_2, \dots, \sigma_{2^j}\}$ . If  $R_i \subseteq \Omega$  is any random set such that  $\Pr[\sigma \in R_i] = 2^{-i}$  for all  $\sigma \in \Omega$  and  $m_i = \max_{\sigma \in R_i} f(\sigma)$ , then

$$\Pr[m_i > b_{i-c}] < |\Omega_{i-c}| 2^{-i} = 2^{-c} \quad (5)$$

In other words, for  $m_i$  to be unlikely to be ‘‘too big’’ it suffices for  $R_i$  to have the right (expected) size, without any requirement of its geometry beyond uniformity. For example,  $R_i$  could even be a random subcube of  $\Omega$ , an extremely computation-friendly constraint: pick  $i$  variables at random and assign them random values.

Achieving (4) is far more subtle. This is because in order for  $\Pr[m_i \geq b_{i+c}]$  to not vanish the random variable  $X_i = |\Omega_{i+c} \cap R_i|$  must be well-behaved. In particular, observe that  $\mathbb{E}X_i = 2^c$  and we aim for  $c$  to be small, e.g.,  $c = 2$ , so the expectation of  $X_i$  is modest. If  $X_i$  realizes its modest expectation via a lottery phenomenon, i.e., typically  $X_i = 0$  but rarely  $X_i$  is very large we are in trouble. To control for this possibility we use the Paley-Zygmund inequality asserting that if  $X$  is any non-negative integer random variable, then  $\Pr[X > 0] \geq (\mathbb{E}X)^2/\mathbb{E}X^2$ . Taking  $R_i$  to be a random cube is thus exposed as a bad idea:

if  $\Omega_{i+c}$  is also a cube, their potential alignment implies  $\mathbb{E}X_i^2 \gg (\mathbb{E}X_i)^2$ .

To minimize  $\mathbb{E}X_i^2$  we would like to find random sets  $R_i$  that behave like ‘‘mists’’, minimizing the probability of having an atypically large intersection with any fixed set. Error-correcting codes are ideal for this, with linear codes particularly so, as they are specified via linear equations. Motivated by these considerations, let

$$R_i = \{\sigma \in \{0, 1\}^n : A\sigma = b\} \ ,$$

where the vector  $b \in \{0, 1\}^i$  is uniformly random, while  $A \in \{0, 1\}^{i \times n}$  is *arbitrary* (even deterministic), for now.

It is easy to see that the uniformity of  $b$  over  $\{0, 1\}^i$  alone suffices to make  $R_i$  uniform over  $\{0, 1\}^n$ , i.e., for all  $\sigma$ ,

$$\Pr[\sigma \in R_i] = 2^{-i} \ . \quad (6)$$

At the same time, for any  $S \subseteq \Omega$ , if  $X_i = |S \cap R_i|$ , then

$$\begin{aligned} \mathbb{E}X_i^2 &= \mathbb{E} \left( \sum_{\sigma \in S} \mathbf{1}_{\sigma \in R_i} \right)^2 \\ &= \sum_{\sigma, \sigma' \in S} \mathbb{E}(\mathbf{1}_{\sigma \in R_i} \mathbf{1}_{\sigma' \in R_i}) \\ &= \mathbb{E}X_i + \sum_{\substack{\sigma, \sigma' \in S \\ \sigma \neq \sigma'}} \Pr[A\sigma = b = A\sigma'] \ . \end{aligned}$$

To deal with the sum above note that, trivially,

$$\Pr[A\sigma = b = A\sigma'] = \Pr[A\sigma = b \wedge A(\sigma - \sigma') = \mathbf{0}] \ .$$

Fix any distinct pair  $\sigma, \sigma'$ . Choosing  $A$  first (to determine if  $A(\sigma - \sigma') = \mathbf{0}$ ) and then choosing  $b$  (to determine if  $A\sigma = b$ ) we see that  $\Pr[A\sigma = b = A\sigma'] = 2^{-i} \Pr[A(\sigma - \sigma') = \mathbf{0}]$ . Therefore, without any assumptions on either the set  $S$  or the distribution of  $A$ , we can conclude that

$$\mathbb{E}X_i^2 = \mathbb{E}X_i + 2^{-i} \sum_{\substack{\sigma, \sigma' \in S \\ \sigma \neq \sigma'}} \Pr[A(\sigma - \sigma') = \mathbf{0}] \ . \quad (7)$$

To move beyond this point we must make some assumptions about (the distribution of)  $A$ . One such assumption, of course, would be that for all  $\sigma - \sigma' = \tau \neq \mathbf{0}$ ,

$$\Pr[A\tau = \mathbf{0}] \leq 2^{-i} \ . \quad (8)$$

It is not hard to see that if (8) holds then:

- (a)  $R_i$  is an  $i$ -thinner.
- (b)  $\mathbb{E}X_i^2 \leq \mathbb{E}X_i + (\mathbb{E}X_i)^2$ . Thus, by the Paley-Zygmund inequality,  $\Pr[m_i \geq b_{i+c}] > 1 - 2^{-c}$ .
- (c) Taking  $c = 2$  and recalling (5) we see that (3), (4) are satisfied with  $\epsilon = 1/4$  (and we have proven Theorem 1).

The above viewpoint exposes how extraordinarily strict is the requirement of universality: it asks that an element  $\tau \in \Omega$  that has a single 1 should be no more likely to solve  $A\tau = \mathbf{0}$  than  $\tau = \mathbf{1}$ . Clearly, this can only be satisfied if the rows of  $A$  have very large expected mass. In [15] the universality requirement was dropped and the case where the entries of  $A$  are i.i.d. Bernoulli taking the value 1 with probability  $p \leq 1/2$  was analyzed. The bound derived for the contribution of each  $\sigma \in S$  to the sum in (7) under this scheme is dominated by its pairing with  $\sigma'$  forming a Hamming ball centered at  $\sigma$ . Note, though, that if  $R_i$  is an error-correcting code and  $\sigma \in R_i$ , then it is extremely unlikely that *any*  $\sigma'$  near  $\sigma$  will also be in  $R_i$ . Indeed, the most basic metric of the quality of an error-correcting code is its distance, i.e., the minimum distance of any two codewords. This “self-repulsive” property of error-correcting codes is our key insight in regards to their statistical properties.

### 5.1 LOW DENSITY PARITY CHECK CODES

In the basic LDPC construction, the (random) system of linear equations is represented as a bipartite graph with variables on the left and equations (checks) on the right, with adjacency denoting entailment, i.e., that the variable participates in the equation. To create a code with  $n$  variables, i.e., with codewords in  $\{0, 1\}^n$ , one first specifies the numbers  $\{\lambda_j\}$  and  $\{\phi_j\}$  of variables and equations, respectively, of each degree  $j$ . In the simplest case,  $\lambda_j = \phi_k = 0$  for all but one value of  $j$  and  $k$ , respectively, i.e., the graph is (bi-)regular. In general, with the degree sequences thus fixed, a random bipartite graph is chosen uniformly at random subject to the degree constraints. This uniformity implies that  $\Pr[A\tau = \mathbf{0}]$  depends only on the weight of  $\tau$ , i.e., its number of 1s, for every  $\tau \in \{0, 1\}^n$ .

An even better construction of LDPC codes than the above is the Progressive Edge Growth (PEG) construction [3]. Its main feature, relative to the standard LDPC construction, is that it tries to maximize the length of the shortest cycle (girth) of the resulting bipartite graph, as short cycles contribute significantly to the formation of small stopping sets. Motivated by these considerations we replaced the dense and sparse parity ensembles with PEG constructed LDPC codes. If  $N(w)$  is the number of codewords of weight  $w$  for a given code, then the probability in (8) is  $P(w) = N(w)/\binom{n}{w}$ . Ideally,  $P(w)$  would be constant for all  $w > 0$ , i.e., (8) would be an equality, which is precisely what happens when  $A \sim \text{Ber}(i \times n)$ . More generally, the flatter  $P(w)$  is, the better the statistical properties of  $R_i$ .

To demonstrate the superiority of PEG LDPC over the sparse ensemble we plot in Figure 3 the empirical value of  $\log P(w)$  for parity matrices of size  $20 \times 40$ , derived by exhaustively enumerating each code’s, roughly,  $2^{40/2} \approx 1$  million codewords (exhaustive enumeration was chosen because the number of codewords can have non-trivial fluctu-

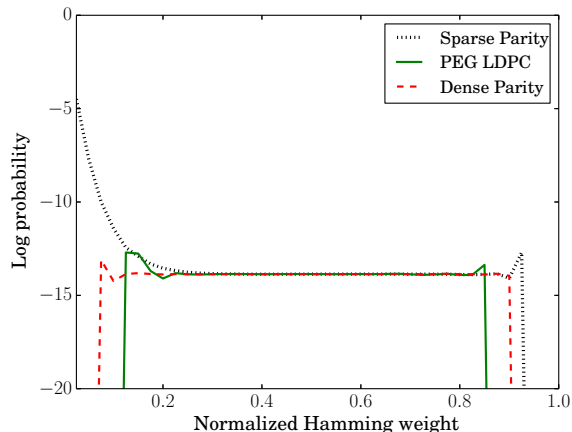


Figure 3: Empirical  $\log P(w)$  for  $w \in [1, 40]$  of dense parity ensemble (average 20 vars/equation), sparse parity ensemble (avg 8 vars/eq), and PEG LDPC (avg 8 vars/eq.)

ations for small  $n$ .) For the dense and sparse parity ensembles we generated 100 matrices each and report the mean; the PEG construction for LDPC is deterministic. We only considered codes with  $n = 40$  variables, as exhaustive enumeration rapidly becomes intractable with  $n$ . Already, though, for  $n = 40$  the behavior is very stable and it is easy to prove that flatness increases as  $n$  grows.

As can be seen, for a wide range of  $w$  both the sparse parity ensemble and the LDPC ensemble match  $P(w)$  perfectly (the fact that dense parity itself is not flat for  $w \notin [3, 36]$  is a finite-size effect). Crucially, though, for small  $w$  there is a big difference, with the sparse parity ensemble containing many more codewords (note that the vertical axis is logarithmic). The over-representation of low-weight codewords causes nearby pairs in  $S \subseteq \Omega$  to contribute disproportionately to the sum in (7), potentially causing the variance of  $X_i$  to blow up if  $S$  is clustered, e.g., if  $S$  is a cube. No such blowup occurs for the PEG codes even for such small  $n$ , witnessing their very good statistical properties. As the study of the codeword weight distribution function of LDPC codes greatly exceeds the scope of this work, we leave a formal proof that thinning by LDPC codes give rise to small-variance estimators as future work.

We claimed earlier that LDPC codes should be far preferable to random parity matrices. To that end we plot the performance of CPLEX on the Ising grid in Figure 4. The only difference between the three plots is in the parity matrix  $A$  used to define  $R_i = A\sigma + b$ . The collapse of CPLEX beyond a certain number of constraints was already pointed out in Figure 1. For LDPC PEG codes no such collapse occurs and CPLEX remains competitive with LocalSolver and random affine maps until  $i \approx 40$  even with a time-out at small as 30 seconds. In contrast, as demonstrated in Figure 5, LocalSolver, unaware of the variable/product

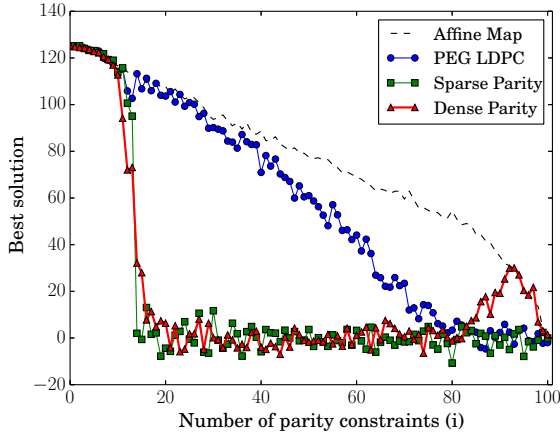


Figure 4:  $10 \times 10$  Ising grid with  $C = 1.0$  and  $F = 0.1$ . *Experiment:* for  $i \in [0, 100]$ , generate  $R_i$  via an  $i \times 100$  parity matrix chosen from 3 different ensembles (see legend); seek  $\max_{\sigma \in R_i} f(\sigma)$  for 30 seconds using CPLEX. *Plot:* for each  $i$  and each ensemble repeat the experiment 10 times and report the binary logarithm of the median value found. (As a yardstick, we also plot the values found by Local Solver when  $R_i$  is a random affine subspace.)

structure of  $\Omega$  (and the factorization of  $f$  over  $\{\psi_\alpha\}$ ) does not “feel” the difference between different parity check matrices, consistent with our hypothesis that it is the presence (and exploitation) of “safe” decoding moves that causes the dramatic improvement in the performance of CPLEX.

## 6 BRANCH-AND-BOUND

Our last observation is that not all  $\hat{b}_i$  are equally important (or even necessary) for an accurate estimate  $\hat{Z}$ . For instance, if  $f(\sigma) \in \{0, 1\}$ , it suffices to find the boundary  $k$  such that  $b_i = 1$  for  $i \leq k$  and  $b_i = 0$  for  $i > k$ . Using binary search we can do this by solving only  $\log n$ , instead of  $n$ , optimization problems as in (2).

To generalize let  $I = \{i_0, i_1, i_2, \dots, i_s\}$  be the set of quantiles estimated so far, where  $0 = i_0 < i_1 < \dots < i_s = n$ . Now define

$$U_I = b_0 + \sum_{j=0}^{s-1} b_{i_j} \left( \sum_{i=i_j}^{i_{j+1}-1} 2^i \right)$$

$$L_I = b_0 + \sum_{j=0}^{s-1} b_{i_{j+1}} \left( \sum_{i=i_j}^{i_{j+1}-1} 2^i \right).$$

By construction,  $U_I \geq U \geq Z \geq L \geq L_I$ . Let  $\hat{U}_I$  and  $\hat{L}_I$  be the estimated counterparts, of  $U_I, L_I$ , respectively, with  $\hat{b}_i$  in place of  $b_i$ . To identify the next quantile to estimate we consider the successive pairs  $(i_\ell, i_{\ell+1})$  in  $I$  and for each such pair  $(i_\ell, i_{\ell+1}) = (i, j)$  we define

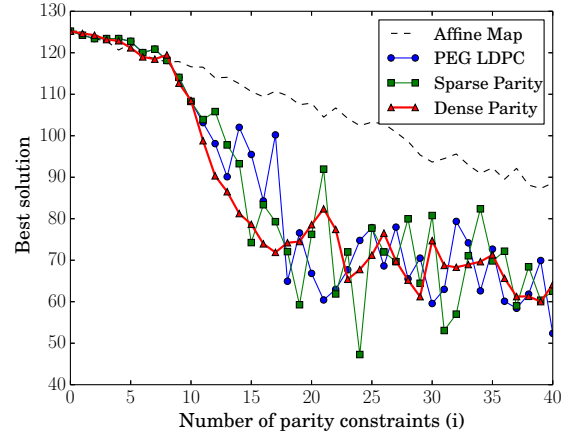


Figure 5:  $10 \times 10$  Ising grid with  $C = 1.0$  and  $F = 0.1$ . *Experiment:* for  $i \in [0, 40]$ , generate  $R_i$  via an  $i \times 100$  parity matrix chosen from 3 different ensembles, or as a random affine subspace (see legend); seek  $\max_{\sigma \in R_i} f(\sigma)$  for 30 seconds using LocalSolver. *Plot:* for each  $i$  and method repeat the experiment 10 times and report the binary logarithm of the median value found.

$\text{Gap}(i, j) = (\hat{b}_i - \hat{b}_j) \sum_{q=i}^j 2^q$ . At each iteration, we chose the pair  $(i, j)$  with maximum gap and estimate  $b_k$ , where  $k = \lfloor \frac{i+j}{2} \rfloor$ . Once the ratio  $\hat{U}_I / \hat{L}_I$  drops below the desired accuracy threshold, the process can stop early (see Table 6 for some indicative results).

---

### Algorithm 1 Branch-and-Bound

---

- 1:  $\hat{b}_0 \leftarrow \text{ESTIMATE}(b_0)$
  - 2:  $\hat{b}_n \leftarrow \text{ESTIMATE}(b_n)$
  - 3:  $I \leftarrow \{0, n\}$
  - 4:  $\hat{U}_I \leftarrow \hat{b}_0 2^n$
  - 5:  $\hat{L}_I \leftarrow \hat{b}_n 2^n$
  - 6: **while**  $(\hat{U}_I > \hat{L}_I \cdot \text{Tolerance})$  **do**
  - 7:     Find successive  $i, j \in I$  maximizing  $\text{Gap}(i, j)$
  - 8:      $k \leftarrow \lfloor (i + j) / 2 \rfloor$
  - 9:      $\hat{b}_k \leftarrow \text{Estimate}(b_k)$
  - 10:     $I \leftarrow I \cup \{k\}$
  - 11:    Compute  $\hat{U}_I, \hat{L}_I$
  - 12: **end while**
  - 13: **Return**  $(\hat{U}_I + \hat{L}_I) / 2$
- 

The benefit of branch and bound is universal, i.e., independent of the  $\{b_i\}$  estimation method. For example, over 24 problems on Ising grids it yielded a 7x average speedup.

| $C$ ( $F = 0.1$ ) | 0.25 | 0.5 | 1.0 | 1.5 | 2 | 2.5 | 3  |
|-------------------|------|-----|-----|-----|---|-----|----|
| Speedup (x)       | 11   | 5   | 3   | 7   | 9 | 11  | 12 |

Table 1: Speedup by Branch and Bound

## 7 EXPERIMENTS

The ferromagnetic Ising grid is a canonical spin glass model. Binary variables (spins)  $x_i \in \{\pm 1\}$  are placed on the vertices of a  $\sqrt{n} \times \sqrt{n}$  grid  $(V, E)$  and have nearest-neighbor interactions and a local (to each spin) field. Thus,

$$f(x) = \prod_{i \in V} \psi_i(x_i) \prod_{(i,j) \in E} \psi_{i,j}(x_i, x_j) , \quad (9)$$

with  $\psi_i(x_i) = \exp(F_i x_i)$  and  $\psi_{i,j}(x_i, x_j) = \exp(C_{i,j} x_i x_j)$ , where the local fields  $F_i$  are i.i.d.  $U[-F, F]$  and the coupling strengths  $C_{i,j}$  are i.i.d.  $U[0, C]$ . (As  $C_{i,j} \geq 0$ , configurations where neighboring spins align are favored.)

The model has been widely used as a test case for partition function estimation [4][8][14][15] due to its flexibility: as  $C$  is increased, the dominant contribution to the partition function shifts from configurations with many unaligned neighbors to configurations with few unaligned neighbors. We focus on the particularly challenging setting  $C \approx 1$ . A side benefit of this choice (not unrelated to hardness) is that a wide range of quantiles contribute significantly to  $Z$ , thus exercising each method for a wide range of thinning.

### 7.1 THE ALGORITHMS

Given the complementary nature of thinning via parity matrices and thinning via affine subspaces, it is natural to combine our two ideas into one algorithm that uses LDPC parity check matrices for small  $i$  (“light” thinning) and random affine subspaces for large  $i$  (“heavy” thinning). To reduce the confounding factors we simply used parity matrices for  $i \in [1, 33]$  and affine subspaces for  $i \in [34, 100]$ .

Besides the junction-tree algorithm used to compute  $Z$  exactly, we also evaluated the Mean Field approximation, and Tree-Reweighted Belief Propagation (TRWBP) [6], providing a lower and an upper bound for the partition function, respectively. We use the libDAI [10] implementations of all three algorithms. The WISH algorithm is the CPLEX implementation by the authors of [14].

### 7.2 THE RESULTS

WISH is not competitive with our algorithm in certain settings, as indicated by Figure 6 (note that the vertical axis is logarithmic). For example, our algorithm achieves better accuracy with a 10-second timeout than WISH achieves with 360 seconds. When this difference in accuracy is combined with the Branch and Bound speedup, our algorithm is over 100x faster than WISH.

In Figure 7 we compare all four algorithms for  $F = 0.1$  and various values of  $C \in [0.25, 3.0]$ . Rather than comparing run times, which in order to be fair would require adapting the timeout of each algorithm to the difficulty it

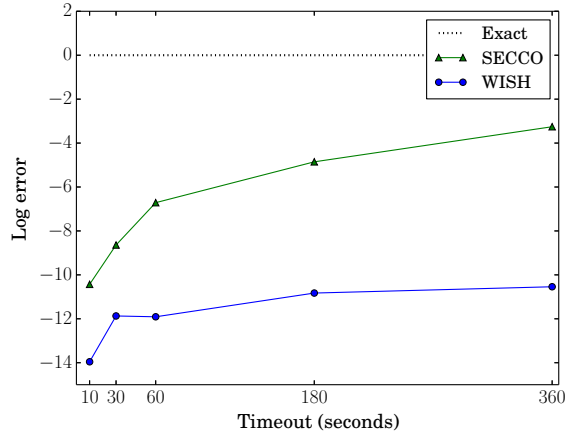


Figure 6:  $10 \times 10$  Ising grid with  $C = 1.0$  and  $F = 0.1$ . *Experiment:* run WISH and our algorithm with a timeout of  $t \in \{10, 30, 60, 180, 360\}$  seconds per instance to get an estimate  $\hat{Z}$  (see legend). *Plot:* report  $\log_2(\hat{Z}/Z)$ .

experiences, we have chosen the more transparent experiment of running each algorithm with the same timeout of 360 seconds across all instances and all  $i \in [n]$  and comparing the accuracy achieved in the final estimate of  $\hat{Z}$ .

The case of high coupling strengths is easy for both algorithms as the dominant contribution to  $Z$  comes from few configurations of high probability and, thus, only light thinning is performed. For  $C \in \{0.5, 0.75, 1.0\}$ , though, our algorithm outperforms WISH by a significant margin (the vertical axis is logarithmic).

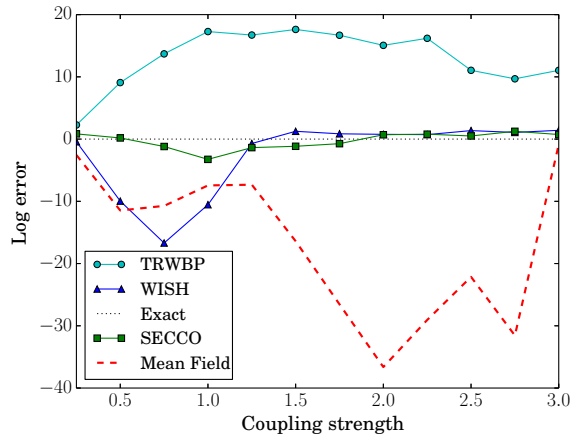


Figure 7:  $10 \times 10$  Ising grid with  $C \in [0.25, 3.0]$ ,  $F = 0.1$ . *Experiment:* for  $C \in \{0.25, 0.50, \dots, 3.0\}$ , determine  $Z$  and run four algorithms to get an estimate  $\hat{Z}$  (see legend). Mean Field and TRWBP are run to termination. WISH and our algorithm have a 360 second timeout for each instance. *Plot:* For  $C \in \{0.25, 0.50, \dots, 3.0\}$  report  $\log_2(\hat{Z}/Z)$ .



## References

- [1] A. Bulatov and M. Grohe. The complexity of partition functions. *Theoretical Computer Science*, 348(2-3):148–186, 2005.
- [2] D. Allouche, S. de Givry, and T. Schiex. Toulbar2, an open source exact cost function network solver. Technical report, Technical report, INRIA, 2010.
- [3] X. Hu, E. Eleftheriou, and D. Arnold. Regular and irregular progressive edge-growth tanner graphs. *IEEE Transactions on Information Theory*, 51(1):386–398, 2005.
- [4] T. Hazan and T. Jaakkola. On the partition function and random maximum a-posteriori perturbations. In *ICML*. icml.cc / Omnipress, 2012.
- [5] M. Jordan, Z. Ghahramani, T. Jaakkola, and L. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, 1999.
- [6] M. Wainwright, T. Jaakkola, and A. Willsky. Tree-reweighted belief propagation algorithms and approximate ml estimation by pseudo-moment matching. In *Workshop on Artificial Intelligence and Statistics*, volume 21, 2003.
- [7] M. Jerrum and A. Sinclair. *Approximation Algorithms for NP-hard Problems*, chapter The Markov Chain Monte Carlo Method: An Approach to Approximate Counting and Integration, pages 482–520. PWS Publishing Co., Boston, MA, USA, 1997.
- [8] T. Hazan, S. Maji, and T. Jaakkola. On sampling from the gibbs distribution with random maximum a-posteriori perturbations. In *NIPS*, pages 1268–1276, 2013.
- [9] T. Benoist, B. Estellon, F. Gardi, R. Megel, and K. Nouioua. Localsolver 1.x: a black-box local-search solver for 0-1 programming. *4OR*, 9(3):299–316, 2011.
- [10] J. Mooij. libDAI: A free and open source C++ library for discrete approximate inference in graphical models. *Journal of Machine Learning Research*, 11:2169–2173, August 2010.
- [11] C. Gomes, A. Sabharwal, and B. Selman. Model counting: A new strategy for obtaining good bounds. In *AAAI*, pages 54–61. AAAI Press, 2006.
- [12] C. Gomes, W. Hovee, A. Sabharwal, and B. Selman. Counting CSP solutions using generalized XOR constraints. In *AAAI*, pages 204–209. AAAI Press, 2007.
- [13] S. Ermon, C. Gomes, A. Sabharwal, and B. Selman. Optimization with parity constraints: From binary codes to discrete integration. In *UAI*. AUAI Press, Corvallis, Oregon, 2013.
- [14] S. Ermon, C. Gomes, A. Sabharwal, and B. Selman. Taming the curse of dimensionality: Discrete integration by hashing and optimization. In *ICML*, volume 28 of *JMLR Proceedings*, pages 334–342. JMLR.org, 2013.
- [15] S. Ermon, C. Gomes, A. Sabharwal, and B. Selman. Low-density parity constraints for hashing-based discrete integration. In *ICML*, volume 32 of *JMLR Proceedings*, pages 271–279. JMLR.org, 2014.
- [16] M. Sipser. A complexity theoretic approach to randomness. In *STOC*, pages 330–335. ACM, 1983.
- [17] M. Thurley. An approximation algorithm for #k-sat. In *STACS*, volume 14 of *LIPICs*, pages 78–87. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2012.
- [18] S. Toda. PP is as hard as the polynomial-time hierarchy. *SIAM Journal on Computing*, 20(5):865–877, 1991.
- [19] L. Valiant and V. Vazirani. Np is as easy as detecting unique solutions. *Theoretical Computer Science*, 47(3):85–93, 1986.

---

# Learning the Structure of Sum-Product Networks via an SVD-based Algorithm

---

**Tameem Adel**  
Radboud University

**David Balduzzi**  
Victoria University of Wellington

**Ali Ghodsi**  
University of Waterloo

## Abstract

Sum-product networks (SPNs) are a recently developed class of deep probabilistic models where inference is tractable. We present two new structure learning algorithms for sum-product networks, in the generative and discriminative settings, that are based on recursively extracting rank-one submatrices from data. The proposed algorithms find the subSPNs that are the most coherent *jointly* in the instances and variables – that is, whose instances are most strongly correlated over the given variables.

Experimental results show that SPNs learned using the proposed generative algorithm have better likelihood and inference results – and also much faster – than previous approaches. Finally, we apply the discriminative SPN structure learning algorithm to handwritten digit recognition tasks, where it achieves state-of-the-art performance for an SPN.

## 1 INTRODUCTION

Sum-product networks (SPNs), introduced in Poon and Domingos [2011], provide compact, tractable representations of probability distributions. Layers of hidden variables are added to the model so long as they maintain compactness whilst keeping inference tractable.

In this work, we present a new SPN structure learning algorithm that constructs an SPN by identifying coherent subSPNs that are sought concurrently across both the instance and variable dimensions. The subSPN search procedure aims at compactly and tractably representing the data and is capable of splitting the data matrix across both dimensions at once, if this leads to a better data representation.

**Contribution.** We make two main contributions. The first is  $\text{SPN-SVD}$ , a new SPN structure learning algorithm

based on rank-one (rank-1) submatrix extraction. The problem of finding subSPNs is reformulated as a problem of finding approximate rank-1 submatrices of the data matrix. An important feature of the approach is that it splits the data along two dimensions (variables and instances) simultaneously when doing so optimises the objective. In contrast, previously developed approaches to learning the structure of SPNs split the data across one dimension only, at a time, without taking into consideration that such local improvement might drift the overall resulting SPN away from the optimal representation.

The second main contribution is an extension of our structure learning algorithm to the setting of *discriminative* learning [Gens and Domingos, 2012]. The discriminative structure learning algorithm,  $\text{DSPN-SVD}$ , first extracts the features that are the most dependent on the labels, where dependence is measured via the Hilbert-Schmidt Independence Criterion (HSIC), and then recursively applies  $\text{SPN-SVD}$ . To the best of our knowledge, it is the first structure learning algorithm designed for discriminatively training SPNs.

The performance of both algorithms is extensively evaluated. When evaluated on the Caltech-101 and Olivetti image datasets,  $\text{SPN-SVD}$  outperforms other SPN algorithms, with higher log-likelihood (LL) values and much faster performance. The discriminative structure learning algorithm achieves state-of-the-art performance on handwritten digit classification when compared with other SPN algorithms.

## 2 SUM-PRODUCT NETWORKS

SPNs are built by composing tractable distributions. A tractable distribution is a distribution whose partition function and mode can be computed in time  $O(1)$  [Gens and Domingos, 2013]. A tractable univariate distribution,  $D_X$ , is an SPN. SPNs provide a representation in which single tractable distributions of the form  $D_X$  are combined into a richer and more complex distribution, provided that the resulting distribution remains tractable.

An SPN is a rooted directed acyclic graph (DAG),  $Gr$ , whose leaves are univariate distributions, and whose internal nodes are sum and product nodes. Edges from a sum node to its children are assigned positive weights,  $W$ . Let  $Br(S)$  denote the branches (children) of  $S$ . The scope,  $sc$ , of a sum-product network,  $S(Gr, W)$ , is the set of variables that appear in the leaf nodes of the SPN [Poon and Domingos, 2011]. Sum nodes are denoted by  $Sum_i(Br_1 : w_1, Br_2 : w_2, \dots)$ , where  $Br_1, Br_2, \dots$  are the branches and  $w_1$  and  $w_2$  are their respective weights. Similarly, product nodes are denoted by  $Prd_i(Br_1, Br_2, \dots)$ .

The two composition rules used in SPNs are defined as follows. Firstly, a product,  $Prd$ , of SPNs,  $sub_1, \dots, sub_k$ , over disjoint scopes, is an SPN, rooted by  $Prd$ :

$$\forall sub_i, sub_j \in Br(Prd) : sc(sub_i) \cap sc(sub_j) = \phi \quad (1)$$

A decomposable SPN is one in which each product node satisfies Eq. (1).

Secondly, a positive weighted sum,  $Sum$ , of SPNs over the same scope is an SPN rooted by  $Sum$  [Gens and Domingos, 2013]:

$$\forall sub_i, sub_j \in Br(Sum) : sc(sub_i) = sc(sub_j) \quad (2)$$

A complete SPN is one in which each sum node satisfies Eq. (2). A sum node,  $Sum$ , can be thought of as the result of summing out a hidden variable. The sum of weights of all the branches of a sum node is always equal to 1 ( $\sum_{sub \in Br(Sum)} w_{sub} = 1$ ).

## 2.1 Related Work on Structure Learning

The emphasis in the SPN literature has recently shifted from learning parameters to learning the structure of models. Parameter learning algorithms assume a fixed structure and learn weights using either generative [Poon and Domingos, 2011] or discriminative [Gens and Domingos, 2012] training. A hard EM algorithm is used by Poon and Domingos [2011] to perform generative parameter learning on SPNs. Deep SPNs are learned successfully using hard EM with a *pre-defined* network structure. A discriminative parameter learning algorithm based on gradient descent was introduced in Gens and Domingos [2012].

The first algorithm to learn the structure of an SPN from data was proposed by Dennis and Ventura [2012]. The algorithm first clusters data instances and creates a corresponding sum node. Then, it clusters variables in a top-down approach, creating product nodes. There are three potential problems with the algorithm. The first is that any context-specific independences that appear after the first clustering are not taken into consideration because instances are not clustered after the first step. Secondly, the algorithm is based on a clustering method that ignores correlation between variables. It will therefore tend to place

dissimilar, but strongly correlated, variables in different clusters. Finally, the structure and weights are learned using two distinct methods.

A bottom-up approach, based on greedily merging small image regions into larger regions, was introduced in Peharz et al. [2013]. An online learning algorithm was proposed by Lee et al. [2013], where the problem was cast as an online clustering problem. They develop an incremental SPN structure learning algorithm based on dynamically modifying the number of clusters based on incoming data.

The most prominent general SPN structure learning algorithm was proposed by Gens and Domingos [2013]. It applies a recursive top-down approach which, at each step, checks whether variables can be split into approximately independent subsets – in which case a product node is constructed. Otherwise, the current instances are clustered, and a sum node is returned with weights proportional to the number of instances in each cluster. The algorithm greedily optimises the log-likelihood and overcomes several limitations in Dennis and Ventura [2012]. However, it only searches locally for the ideal splitting candidate at each step, see discussion of Table 1 below.

To the best of our knowledge, the most recent algorithm for general SPN structure learning was proposed in Rooshenas and Lowd [2014]. The authors adapt a method based on mixture modelling and arithmetic circuit learning. It does not only apply local modifications in the search of an optimal model as arithmetic circuit learning could lead to global changes. However, a global search over the data is neither systemic nor guaranteed.

Peharz et al. [2014] learn the structure of a restricted class of SPNs, where each sum node can have no more than one branch with a non-zero output for a certain input. Nath and Domingos [2014] develop an algorithm that learns the structure of relational SPNs.

Our focus is on learning the structure of general SPNs in both the generative and discriminative settings.

## 3 LEARNING THE STRUCTURE OF AN SPN

Previous SPN structure learning algorithms cluster instances without ensuring that the clustering respects context-specific independences (independences that hold only among instances of a specific context or cluster). In contrast, our proposed algorithm (*SPN-SVD*) concurrently checks the data matrix across both the instance and variable dimensions, looking for coherent subSPNs in the form of rank-1 submatrices.

**Motivating Example.** It is useful to consider a simple example in detail, so as to understand how *SPN-SVD* differs from *SPN-Gens*. Tables 1 & 2 contrast the steps

taken by SPN-SVD and SPN-Gens [Gens and Domingos, 2013]. Table 1(A) shows the data matrix, which consists of 6 instances with 4 variables each. SPN-Gens clusters the data into 2 clusters, as shown in Table 1(B) and in Figure 1. In the example, the cluster of elements with value 18 is split.

Table 1: SPN-Gens on an Example Data Matrix. Rows are instances and columns are variables.

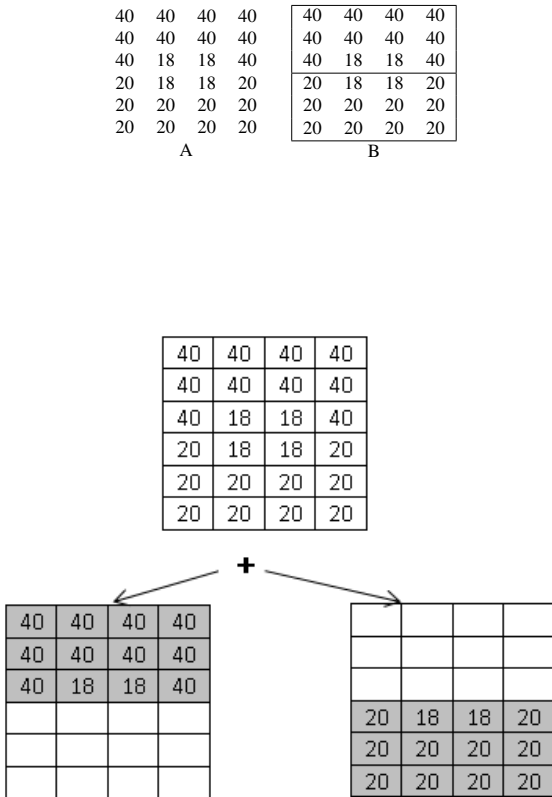


Figure 1: The SPN Structure Learned from Table 1 by SPN-Gens.

SPN-SVD deals with the same data quite differently, as shown in Table 2. The algorithm simultaneously searches over instances and variables and therefore immediately identifies the submatrix with all entries equal to 18, chooses it as a rank-1 submatrix. The algorithm then decomposes the original matrix into three components and acts recursively on each.

The crucial difference between the two algorithms is that SPN-SVD identifies the submatrix of entries with value 18 as an “atom” in the data, resulting in a graph structure that captures an important feature of the data, whereas SPN-Gens does not. Quantitatively, the final LL value for SPN-Gens is  $-2$  whereas for SPN-SVD it is  $-1.39$ .

Table 2: SPN-SVD Applied to the Matrix from Table 1.

|    |    |    |    |    |    |    |    |
|----|----|----|----|----|----|----|----|
| 40 | 40 | 40 | 40 | 40 | 40 | 40 | 40 |
| 40 | 40 | 40 | 40 | 40 | 40 | 40 | 40 |
| 40 | 18 | 18 | 40 | 40 | 40 | 18 | 18 |
| 20 | 18 | 18 | 20 | 20 | 18 | 18 | 20 |
| 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 |
| 20 | 20 | 20 | 20 | 20 | 20 | 20 | 20 |

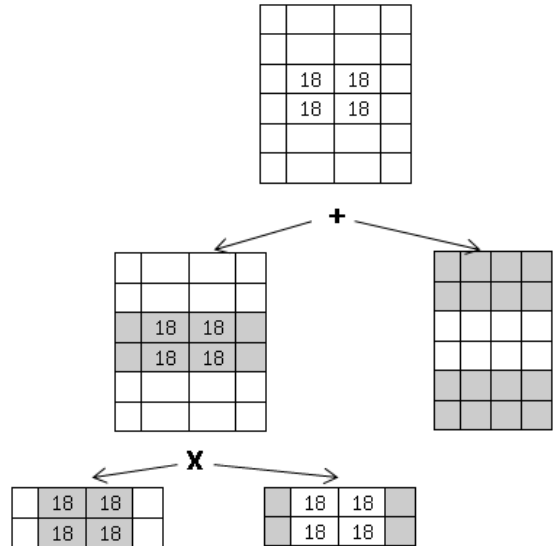


Figure 2: The SPN Structure Learned from Table 1 by SPN-SVD.

In essence, SPN-SVD and SPN-Gens are motivated by two different extreme cases.

SPN-Gens is inspired by the observation that if variables are independent, then they can be decomposed into separate (branches or) leaves of an SPN product node.

In contrast, SPN-SVD is inspired by a complementary observation: if a subset of variables is perfectly correlated over a subset of data, then it forms a rank-1 submatrix. These rank-1 submatrices are the “atoms” out of which SPN-SVD builds an SPN. Whereas SPN-Gens searches for independencies; SPN-SVD searches for correlated components.

Searching for rank-1 submatrices instead of independent variables has three potential advantages. Firstly, correlations are easier to estimate than independence. Secondly, the search for rank-1 submatrices occurs jointly over variables and instances, whereas clustering and identifying independencies are two unrelated procedures. Thirdly, extracting correlated submatrices reduces redundant compu-

tations, resulting in a faster algorithm, see Lemma 1 below.

### 3.1 Extracting a Rank-1 Submatrix

The main subroutine of SPN-SVD is a rank-1 extraction algorithm based on singular value decomposition (SVD). The approach derives from an algorithm for nonnegative matrix factorization (NMF) developed in Biggs et al. [2008a].

Let  $X \in \mathbb{R}^{m \times n}$  denote a data matrix containing  $m$  instances and  $n$  variables. We denote by  $X_{i\bullet}$  the  $i^{\text{th}}$  row of the matrix, corresponding to the  $i^{\text{th}}$  instance of the data, and by  $X_{\bullet j}$  the  $j^{\text{th}}$  column of the matrix, corresponding to the  $j^{\text{th}}$  variable.

In the generative training case, assume the labels, if provided, are included in  $X$ . We introduce the notation  $X_{(M,N)}$  to refer to the *submatrix* of  $X$  consisting of rows  $M \subset \{1, \dots, m\}$  and columns  $N \subset \{1, \dots, n\}$ .

The algorithm extracts the submatrix of the data matrix  $X$  that is closest to having rank-1, denoted by  $B_1$ , by maximising

$$B_1 := \operatorname{argmax}_{X_{(M,N)}} \|X_{(M,N)}\|_F^2 - \gamma \|X_{(M,N)} - \sigma uv^\top\|_F^2, \quad (3)$$

where  $\sigma$  is the maximum singular value of the submatrix  $X_{(M,N)}$  and  $u \in \mathbb{R}^m, v \in \mathbb{R}^n$  are the dominant singular vectors of  $X_{(M,N)}$ , and  $\|\bullet\|_F$  is the Frobenius norm. Recall that the Frobenius norm is the root sum of the squared singular values.

The second term in Eq. (3) thus encourages the optimization to find a submatrix that is close to rank-1. The first term ensures the submatrix is biased towards having large singular values;  $\gamma$  controls the penalty incurred as  $X_{(M,N)}$  deviates from being rank-1.

Algorithm 1 details the subroutine, `extractR1`, used to extract approximate rank-1 submatrices. For fixed  $M$  and  $N$ , `extractR1` exactly coincides with the SVD power method. However, instead of fixing  $M$  and  $N$ , the inner loop searches for the submatrix with the closest rank-1 approximation. Lines 6 and 8 show a heuristic used to solve the NP-hard problem defined in Eq. (3) [Biggs et al., 2008a]. The criterion in lines 6 and 8 of Algorithm 1 decides whether or not to include one column or row separately. This makes the subroutine parallelisable and highly scalable in terms of memory and processing power. `extractR1` computes the *dominant* singular vectors of a *submatrix*, which are less prone to perturbations by noise than the full matrix [Biggs et al., 2008b].

### 3.2 Generative SPN Structure Learning Algorithm (SPN-SVD)

SPN-SVD recursively extracts ‘‘atomic’’ submatrices from the input matrix. Each extraction breaks the input ma-

---

#### Algorithm 1 Function `extractR1(X)`

---

**Input:**  $X \in \mathbb{R}^{m \times n}, \gamma > 1$

**Output:**  $[M, N, Stop]$

- 1: Select  $j_0 \in \{1, \dots, n\}$  to maximise  $\|X_{(:,j_0)}\|_F$
  - 2:  $M = \{1, 2, \dots, m\}, N = \{j_0\}$
  - 3:  $u = X_{(:,j_0)}$
  - 4: **repeat**
  - 5:    $v = X_{(M,:)} \cdot \frac{u(M)}{\|u(M)\|_F}$
  - 6:    $N = \{j : \gamma v(j)^2 - \|X_{(M,j)}\|_F^2 > 0\}$
  - 7:    $u = X_{(:,N)} \cdot \frac{v(N)}{\|v(N)\|_F}$
  - 8:    $M = \{i : \gamma u(i)^2 - \|X_{(i,N)}\|_F^2 > 0\}$
  - 9: **until**  $M, N, u, v$  do not change
  - 10: **if**  $X_{(M,N)} = X$  or  $(|M| = 0$  and  $|N| = 0)$  **then**
  - 11:    $Stop = true$
  - 12: **else**
  - 13:    $Stop = false$
  - 14: **end if**
- 

trix into three pieces, which are glued together as sum and product nodes.

Algorithm 2 details the main steps of SPN-SVD. Rows are instances and columns are variables. Each variable is divided by its standard deviation before extracting rank-1 submatrices. The normalised values are used in the subroutine `extractR1` only, and are not returned or updated in the matrix. Normalisation helps discover correlated sets of variables.

Given an input matrix, subroutine `extractR1` recursively extracts an approximate rank-1 submatrix  $B_1$ . The matrix is then split into three components: the submatrix  $B_1$ , submatrix  $B_2$  consisting of other variables on the same instances as  $B_1$ , and finally submatrix  $B_3$  consisting of the remaining instances.

The optimization in (3) ensures that variables in  $B_1$  are maximally correlated and the remaining variables, captured by  $B_2$ , are largely uncorrelated with  $B_1$ . The algorithm therefore combines  $B_1$  and  $B_2$  via a product node

$$Prd(B_1, B_2).$$

Finally, the remaining *instances*, captured by  $B_3$ , are added via a sum node

$$Sum(B_3 : w_1, Prd(B_1, B_2) : w_2),$$

where  $w_2 = \frac{|M|}{m}$  and  $w_1 = 1 - w_2$ .

The algorithm proceeds recursively by feeding  $B_1, B_2$  and  $B_3$  back into the algorithm as input matrices until one of three base cases is reached:

1. *The input matrix contains a single variable:*

The remaining vector represents a univariate distribution and a leaf node is created (line 2).

---

**Algorithm 2** Function SPN-SVD( $A$ )

---

**Input:**  $A \in \mathbb{R}^{m \times n}, \gamma > 1$

**Output:** Sum-product network  $S$  representing  $A$

- 1: **if** #columns( $A$ ) = 1 **then**
  - 2:   **return** univariate distribution on variable.
  - 3: **else if** #rows( $A$ ) = 1 **then**
  - 4:   **return** return  $Prd$ (variables in  $A$ ).
  - 5: **end if**
  - 6:  $[M, N, Stop] = \text{extractR1}(A, \gamma)$
  - 7: Set  $B_1 = A_{(M, N)}$ ,  $B_2 = A_{(M, N^c)}$  and  $B_3 = A_{(M^c, \{1 \dots n\})}$
  - 8: **if**  $Stop = \text{true}$  **then**
  - 9:   **return** multivariate distribution  $MVLN(A)$
  - 10: **else**
  - 11:   Construct  $Prd(B_1, B_2)$
  - 12:   Call SPN-SVD( $B_1$ ) and SPN-SVD( $B_2$ )
  - 13:   Construct
- $$Sum \left( B_3 : \frac{m - |M|}{m}, Prd(B_1, B_2) : \frac{|M|}{m} \right)$$
- 14:   Call SPN-SVD( $B_3$ )
  - 15: **end if**
- 

2. *The input matrix contains a single instance:*

All variables are independent and a product node is created (line 4). Its leaves are the relevant variables.

3. *The entire input matrix is extracted:*

The variables in  $A$  are highly correlated since  $A$  has rank-1. The algorithm therefore constructs a sum node with a branch per instance in  $A$ , followed by product nodes over the variables. We refer to the node as a *multivariate leaf node* or *MVLN*.

More precisely, if  $A \in \mathbb{R}^{M \times N}$  and  $B_1 \equiv A$  then for each instance  $j$  form product node

$$r_j := Prd_j(A_{j1}, \dots, A_{jN}).$$

Combine the product nodes by summing over instances:

$$MVLN(A) := Sum(r_1 : w, \dots, r_M : w), \quad (4)$$

where  $w = \frac{1}{|M|}$ .

Variables in a rank-1 submatrix cannot be independent. SPN-SVD achieves significant speedups by avoiding redundant searches for independencies and returning MVLNs. In contrast, when SPN-Gens encounters a rank-1 submatrix, it recursively searches for independencies and clusters across its subsets, which leads to a slower implementation and also an SPN with a more complicated structure. The speedup from using MVLNs is reported in the experimental results below.

A commonly used assumption is that data is clustered in strongly correlated groups. For example, algorithms such as the group Lasso seeks solutions where groups of variables are zero together [Bach, 2008]. The following simple Lemma illustrates how large MVLNs arise in the more general setting where groups of variables receive the same, or even approximately the same, values.

**Lemma 1.** *Let  $X \in \mathbb{R}^{m \times n}$  be a data matrix consisting of  $m$  instances. Suppose that  $X$  contains a group of instances  $G \subset \{1, \dots, m\}$  with similarity pattern  $\mathbf{K}_G = \{k | X_{ik} = X_{jk} \text{ for all } i, j \in G\}$ . Then SPN-SVD will find a multivariate leaf node satisfying*

$$|MVLN| \geq |\mathbf{K}_G| \cdot |G|.$$

*Proof.* The result follows immediately since the algorithm will either find the MVLN corresponding to the group, or a larger MVLN.  $\square$

An interesting question, deferred to future work, is to characterise the *collections* of groups with similarity patterns that are best suited to the SPN-SVD algorithm. It is also worth investigating how robustly MVLNs are extracted in the presence of noise.

### 3.3 Discriminative SPN Structure Learning Algorithm (DSPN-SVD)

Finally, we consider the setting of discriminative learning, where the algorithm is provided with labeled data. Discriminative learning models the conditional distribution  $P(Y|X)$ , rather than the joint distribution  $P(X, Y)$ . Discriminative SPNs combine the flexibility of selecting/extracting relevant features, with the tractability and representational prowess of SPNs. They can achieve high classification or regression accuracy by selecting variables that are dependent on  $Y$ . They were first introduced by Gens and Domingos [2012], where parameters were learned on a pre-defined structure.

We propose a new discriminative SPN structure learning algorithm, referred to as DSPN-SVD. We assume the labels are discrete, and belong to the set  $\mathcal{C} = \{1, \dots, l\}$ . The algorithm extracts features  $Z$  from the input matrix,  $X$ , that are maximally correlated (in a suitable sense) with the labels  $Y$ . The algorithm then applies SPN-SVD to the learned features to construct a collection of generative SPNs, one per conditional distribution  $P(Z|Y = j)$  for  $j \in \mathcal{C}$ , that are combined by a single sum-node.

Extracting  $Z$  requires a measure of dependence between variables. We use the Hilbert-Schmidt independence criterion (HSIC), which we briefly recall [Gretton et al., 2005].

Let  $k(x, x')$  and  $l(y, y')$  be kernels on the input space  $\mathcal{X}$  and the label space  $\mathcal{Y}$ , with corresponding feature maps  $\phi :$

$\mathcal{X} \rightarrow \mathcal{F}$  and  $\psi : \mathcal{Y} \rightarrow \mathcal{G}$  respectively. The Hilbert-Schmidt Independence Criterion is

$$HSIC(k, l, P_{XY}) := \|C_{xy}\|_F, \text{ where}$$

$$C_{xy} := \mathbf{E}_{(x,y) \sim P} [(\phi(x) - \mu_x) \otimes (\psi(y) - \mu_y)]$$

is the cross-covariance operator [Fukumizu et al., 2004]. We apply the HSIC for supervised feature selection following Song et al. [2007].

Let  $\Pi := \{W \in \mathbb{R}^{n \times d} : \langle W_{\bullet i}, W_{\bullet j} \rangle = \delta_{ij}\}$  denote the set of orthogonal projections from  $\mathbb{R}^n = \mathcal{X}$  to  $\mathbb{R}^d$ . Given the standard dot product  $\langle \bullet, \bullet \rangle$  on  $\mathbb{R}^d$ , each projection induces a kernel  $K_W(x, y) := \langle Wx, Wy \rangle$  on  $\mathcal{X}$ .

Let  $L(y, y') = \delta_{y, y'}$  be the Kronecker kernel, which is 1 if  $y = y'$  and 0 otherwise. The Kronecker kernel is suitable for the categorical variable,  $Y \in \mathcal{C}$ , because it expresses precisely whether or not two labels are equal. It is easy to extend to real-valued or structured labels by employing more sophisticated kernels.

Let  $X \in \mathbb{R}^{m \times n}$  be a data matrix with labels  $Y \in \mathcal{Y}^m$ , yielding empirical distribution  $\hat{P}_{XY}$ . Construct the centering matrix  $H = (Id_m - m^{-1}11^\top)$  and empirical Kronecker kernel  $L_{ij} = \delta_{y_i = y_j}$ .

**Lemma 2.** Let  $V \in \mathbb{R}^{n \times d}$  be the top  $d$  eigenvectors of

$$\chi := X^\top H L H X \quad (5)$$

Then  $V$  maximizes the Hilbert-Schmidt dependence:

$$V = \operatorname{argmax}_{W \in \Pi} HSIC(K_W, L, \hat{P}_{XY}).$$

*Proof.* The vectors  $V_{\bullet j}$ ,  $j = 1, \dots, d$ , are the eigenvectors of  $\chi$ . They therefore maximize the trace

$$\operatorname{argmax}_V \operatorname{tr}(V^\top X^\top H L H X V)$$

Since  $\operatorname{tr}(AB) = \operatorname{tr}(BA)$ , the objective can be rewritten as:

$$\operatorname{argmax}_V \operatorname{tr}(H X V V^\top X^\top H L) \quad (6)$$

Following Barshan et al. [2011], let  $K = X V V^\top X^\top$ . The objective in Eq. (6) is then

$$\operatorname{tr}(H K H L),$$

which is the HSIC [Gretton et al., 2005].  $\square$

The higher the HSIC value, the stronger the dependence between the projected representation of the data  $Z = X V$  and  $Y$ , and thus the more useful the representation is for discriminative learning.

---

### Algorithm 3 Function DSPN-SVD

---

**Input:**  $X \in \mathbb{R}^{m \times n}, Y \in \mathcal{Y}^m, \gamma > 1, d > 1$

**Output:** Sum-product network  $S$  representing  $Y|X$

- 1: Construct kernel matrix  $L_{ij} = (\delta_{y_i = y_j})_{i,j=1}^m$  and centering matrix  $H = (Id_m - m^{-1}11^\top)$ .
  - 2: Compute the  $d$  eigenvectors  $V \in \mathbb{R}^{n \times d}$  of  $\chi = X^\top H L H X$  with the largest eigenvalues.
  - 3: Set feature matrix  $Z_{m \times d} \leftarrow X_{m \times n} \cdot V_{n \times d}$
  - 4: Construct sum node  $Sum_1(Br_j : w_j)$ , where  $Br_j$  contains all instances with label  $j$ , and weight  $w_j = \frac{m}{\#instances \text{ labeled } j}$ .
  - 5: **for** label  $j$  in  $\mathcal{Y}$  **do**
  - 6:   SPN-SVD( $Br_j, \gamma$ )
  - 7: **end for**
- 

After extracting the features  $Z$ , there are two remaining steps. The first step constructs the base node for the discriminative SPN as a sum node that separates instances belonging to different labels. Nodes in the sum are weighted by the number of instances. The resulting network is thus automatically biased towards more common labels. The second step applies the generative SPN-SVD algorithm to each branch of the sum node using the extracted features in  $Z$ .

The main steps of DSPN-SVD are shown in Algorithm 3.

Figure 3 shows an example with 2 labels and 3 variables. As shown in the tables at the top of the figure, the  $X$  variables are replaced by a more suitable representation,  $Z$ . The base sum node then places instances of each label on separate branches, where each branch's subSPN is in turn learned by SPN-SVD. A simplified example of the feature extraction process is shown in Figure 4. Since the first two features convey no information about the labels, extracting only the third feature,  $Z = X_{\bullet 3}$ , maximises the HSIC.

## 4 EXPERIMENTS

### 4.1 Generatively Trained SPNs

Our main evaluation of SPN-SVD is based on comparing its accuracy and speed to other SPN structure learning algorithms. Following prior work on structure learning [Gens and Domingos, 2013, Rooshenas and Lowd, 2014], we report accuracy in terms of the test-set log-likelihood (LL) and query conditional log-likelihood (CLL). These values are obtained from experiments on the Caltech-101 dataset [Fei-Fei et al., 2004], the Olivetti face dataset [Salakhutdinov and Hinton, 2009], and 20 binary datasets. Caltech-101 is one of the most commonly used image datasets. It contains images divided into 101 categories, e.g. airplanes, cameras and faces. Each object category contains from 40 to 800 images. Images in Caltech-101 are  $64 \times 64$  pixels. The Olivetti dataset contains 400 face images of  $64 \times 64$

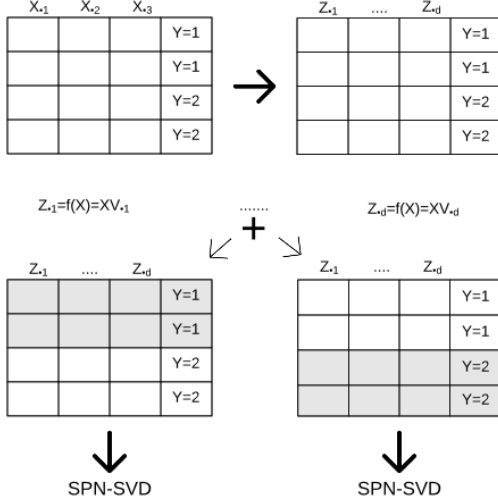


Figure 3: The Discriminative SPN Prior to Running SPN-SVD on Each Sum Node Branch.

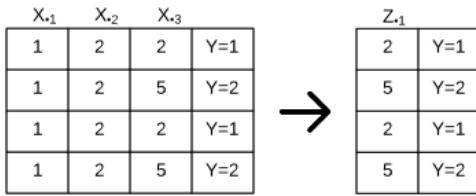


Figure 4: An Example of  $X$ ,  $Y$  and  $Z$  where  $m = 4, n = 3, d = 1$ .  $Z = XV = X[0 \ 0 \ 1]^T$ .

pixels. Importantly, these datasets are not binary, in contrast to previous datasets used for SPN structure learning. The datasets are discrete-valued. Extending to the continuous case is straightforward. 60% of the instances of each object category are used for training, 10% for validation (needed mainly for WinMine) and 30% for testing.

**Accuracy.** For Caltech-101, values of LL and inference are displayed as average values across all object categories, as well as averages for some of the individual object categories, while only the grand average is displayed for Olivetti. By “average”, we mean that the LL values displayed represent their respective summation of LL values divided by the number of test instances. Univariate leaf distributions are multinomials with Laplace smoothing (add 0.1).

We compared SPN-SVD with four algorithms: (1) SPN-Gens [Gens and Domingos, 2013], with code available online; (2) ID-SPN [Rooshenas and Lowd, 2014] by the Libra toolkit; (3) SPN-Dennis [Dennis and Ven-

tura, 2012], which was implemented via algorithms 1, 2 & 3 of Dennis and Ventura [2012]; (4) Bayesian network structure learning with the WinMine toolkit [Chickering, 2002], which was chosen because it can express context-specific independence.

Table 3 shows the test-set LL values obtained for 18 example object categories from Caltech-101, the whole Caltech-101 dataset and the Olivetti face dataset using the SPN-SVD, SPN-Gens, ID-SPN, SPN-Dennis algorithms and WinMine. The greater the LL, the better. The total number of instances (training + test + validation) in each category or dataset is shown in Table 3. Bold red signifies that an algorithm is significantly better than competitors on a category, whereas bold black indicates that an algorithm is better than competitors on a category. Significant results are identified using a paired t-test (performed in the log scale) with  $p = 0.05$ . Out of the 101 Caltech-101 categories, SPN-Gens is significantly better than its competitors in 5 categories, WinMine in 9 categories, ID-SPN in 12 categories while SPN-SVD is significantly better in 42 categories, 9 of which are shown in Table 3.

**Training time.** An important advantage of SPN-SVD is its rapid training time. SPN-SVD took 2.5 hours with 1 CPU to build the SPN and calculate the test-set LL values for Caltech-101 and Olivetti. In contrast, SPN-Gens took 13.5 hours, ID-SPN took 12 hours, and SPN-Dennis took 7.5 hours to perform the same task. WinMine took 2.5 hours to build the Bayesian network and calculate LL values.

Per the discussion of Lemma 1, we expect that larger *MVLNs* lead to larger reductions in run-time. Our experiments show a 3-fold speedup when comparing SPN-SVD’s performance with and without *MVLNs*. Returning *MVLNs* thus accounts for most of the 4.5-fold speedup of SPN-SVD compared to SPN-Gens.

Another major advantage of SPN-SVD is that it has few tuning parameters. The generative algorithm has one parameter,  $\gamma$ , which controls the penalty for deviating from rank-1, whereas DSPN-SVD has a  $2^{nd}$  parameter:  $d$ , the number of extracted features. In comparison, ID-SPN, for example, has:  $L_1$  prior parameters  $C^{ji}$ , split penalty  $SP^{ji}$ , maximum edges  $ME^{ji}$  for each AC node, cluster penalty, standard deviation of the Gaussian priors, and the number of main iterations [Rooshenas and Lowd, 2014].

**Queries.** Next, we investigate the accuracy and speed of queries. Queries are generated following Gens and Domingos [2013]. Experiments are performed with a range of query and evidence variables, see Table 4. A number of instances are selected randomly from the test-set of each object category or dataset, and then queries  $P(Q = q|E = e)$  are created by randomly picking proportions of the variables. The average CLL  $\log P(Q = q|E = e)$  is com-



Table 3: Test-set LL and Learning Time. Results are shown for 18 Caltech-101 Categories, Caltech-101 & Olivetti. Bold red signifies that an algorithm significantly outperforms the rest.

| Dataset              | # inst. | SPN-SVD         | SPN-Gens        | SPN-Dennis | ID-SPN          | WinMine         |
|----------------------|---------|-----------------|-----------------|------------|-----------------|-----------------|
| Faces                | 435     | <b>-1122.71</b> | -1520.03        | -1607.8    | -1440.84        | -1309.37        |
| Faces-Easy           | 435     | <b>-1002.11</b> | -1298.59        | -1490.21   | -1314.09        | -1320.87        |
| Accordion            | 55      | <b>-974.93</b>  | -1114.05        | -1507.79   | -1300           | -1240           |
| Airplanes            | 800     | <b>-587.4</b>   | -920.69         | -1000.3    | -898.7          | -914.81         |
| Anchor               | 42      | -1315.71        | -1420.1         | -1392.28   | -1404.12        | <b>-1239.8</b>  |
| Ant                  | 42      | <b>-770.2</b>   | -1535.82        | -1980.3    | -1264.1         | -1271.94        |
| Background-Google    | 467     | <b>-1105.49</b> | -1316.8         | -2020.88   | -1291.16        | -1220           |
| Barrel               | 47      | <b>-774.23</b>  | -1330.4         | -1289.4    | -1259.7         | -1300.86        |
| Bass                 | 54      | <b>-1051.7</b>  | -1293.11        | -1712.84   | -1321.49        | -1212.37        |
| Beaver               | 46      | -1167.33        | -1570.26        | -1487.79   | -1290.1         | <b>-1012.03</b> |
| Binocular            | 33      | <b>-907.48</b>  | -1390.3         | -1600.3    | -1400.44        | -1309.4         |
| Bonsai               | 128     | <b>-887.42</b>  | -1551.09        | -1979.26   | -1302.37        | -1336.28        |
| Brain                | 98      | -1270.1         | <b>-1208.41</b> | -1498      | -1307.12        | -1286.2         |
| Brontosaurus         | 43      | <b>-837.02</b>  | -1288.13        | -1600.26   | -1393.9         | -1410.61        |
| Buddha               | 85      | -1291.15        | -1374.12        | -1230.8    | <b>-1172.28</b> | -1219           |
| Butterfly            | 91      | <b>-1020.67</b> | -1397.19        | -1535.91   | -1230.11        | -1207.44        |
| Camera               | 50      | -1201.8         | -1470.25        | -1488.85   | <b>-1019.51</b> | -1200.49        |
| Cannon               | 43      | <b>-956.47</b>  | -1303.1         | -1404.71   | -1307.8         | -1288.1         |
| Caltech-101 (All)    | 9144    | <b>-892.93</b>  | -1492.12        | -1780.5    | -1250.6         | -1269.29        |
| Olivetti             | 400     | <b>-189.81</b>  | -294.36         | -302.55    | -295.81         | -293.9          |
| <b>Learning Time</b> |         | 2.5 hours       | 13.5 hours      | 7.5 hours  | 12 hours        | 2.25 hours      |

puted and normalised by the number of query variables following Gens and Domingos [2013]. Table 4 shows the results, for varied proportions of evidence and query variables, in the form of the average CLL for both SPN-SVD and SPN-Gens.

Both SPN-SVD and SPN-GENS achieve average dataset CLL values that are significantly higher than the results obtained by SPN-Dennis, ID-SPN and the conditional marginal likelihood (CMLL) values of WinMine. The latter three results are therefore not reported to save space. Similarly, we only show queries of 5 object categories, rather than 18, along with the average CLL of the whole Caltech-101 and Olivetti datasets.

Across all proportions of object categories, there are 84 categories in which SPN-SVD significantly outperforms SPN-Gens, and 18 where the converse occurs. As inference is linear in the number of edges of an SPN, there is not a major difference between average query time for SPN-SVD and SPN-Gens.

**Image completion.** To confirm that the LL values are visually meaningful, an image completion task was applied to a select few images from Caltech-101. Two images are shown in Figure 5, taken from one of the face categories of Caltech-101, referred to as Faces-easy. The left half of each test image is inferred after building an SPN using training images from the same faces category. In each case, the right half of the test image is given as evidence and the left half is regarded as query variables, and inference is performed by the SPN. The top part of Figure 5 displays the original images and the bottom shows the images inferred

by SPN-SVD.

**Binary datasets.** In Table 5, we report the test-set LL values of SPN-SVD, SPN-Gens and ID-SPN (LL values of SPN-Dennis are significantly lower) on 20 binary datasets used in Gens and Domingos [2013], Rooshenas and Lowd [2014]. The number of instances in a binary dataset ranges from 2k to 388k, and the number of variables ranges from 16 to 1556 [Gens and Domingos, 2013]. Out of the 20 datasets, SPN-SVD outperforms the alternatives in 7 datasets, whereas ID-SPN outperforms the rest in 6 datasets. Significant results are identified using a paired t-test with  $p = 0.05$ .

The results on discrete and binary datasets indicate that SPN-SVD achieves, by far, state-of-the-art performance for an SPN on discrete datasets. This is where interpreting correlations makes a huge difference. SPN-SVD is also at par with SPN state-of-the-art on binary datasets.

## 4.2 Discriminatively Trained SPNs

We present results obtained by applying discriminative SPNs on two handwritten digit recognition datasets, USPS [Hull, 1994] and MNIST [LeCun et al., 1998]. USPS consists of 1100 images per digit for each of the 10 digits. Each image is  $16 \times 16$ . For each digit, 800 images are assigned to the training set and 300 to the test set. MNIST consists of 6000 training images per digit, each of size  $28 \times 28$ , and a test set of 1000 images per digit. Discriminative SPN structures were learned by DSPN-SVD in both cases. The number of extracted features  $d$  was chosen by cross-validation.

Table 4: Average CLL & Query Time. Results are normalised by number of query variables. Results are shown for 5 Caltech-101 categories, Caltech-101 & Olivetti. SVD refers to SPN-SVD, and Gens to SPN-Gens.

| Dataset                | 30% Q., 50% Ev. |               | 10% Q., 30% Ev. |        | 30% Q., 30% Ev. |        | 50% Q., 30% Ev. |        |
|------------------------|-----------------|---------------|-----------------|--------|-----------------|--------|-----------------|--------|
|                        | SVD             | Gens          | SVD             | Gens   | SVD             | Gens   | SVD             | Gens   |
| Faces                  | <b>-0.301</b>   | -0.318        | <b>-0.81</b>    | -0.96  | <b>-0.221</b>   | -0.319 | <b>-0.4</b>     | -0.53  |
| Faces-Easy             | <b>-0.118</b>   | -0.16         | <b>-0.86</b>    | -0.908 | <b>-0.238</b>   | -0.318 | <b>-0.511</b>   | -0.543 |
| Accordion              | -0.314          | <b>-0.312</b> | <b>-0.88</b>    | -0.95  | <b>-0.284</b>   | -0.313 | <b>-0.47</b>    | -0.523 |
| Airplanes              | <b>-0.211</b>   | -0.221        | <b>-0.058</b>   | -0.074 | <b>-0.202</b>   | -0.222 | <b>-0.309</b>   | -0.371 |
| Anchor                 | <b>-0.301</b>   | -0.419        | <b>-0.761</b>   | -0.944 | <b>-0.256</b>   | -0.331 | <b>-0.501</b>   | -0.569 |
| Caltech-101 (All)      | <b>-0.117</b>   | -0.24         | <b>-0.131</b>   | -0.204 | <b>-0.204</b>   | -0.34  | <b>-0.312</b>   | 0.423  |
| Olivetti               | <b>-0.27</b>    | -0.289        | <b>-0.205</b>   | -0.234 | <b>-0.439</b>   | -0.472 | <b>-0.466</b>   | 0.513  |
| <b>Avg. query time</b> | 31 ms           | 30 ms         | 29 ms           | 28 ms  | 30 ms           | 32 ms  | 26 ms           | 27 ms  |

Table 5: Test-set LL for 20 Binary Datasets. Bold red: an algorithm significantly outperforms the rest.

| Dataset     | SPN-SVD       | SPN-Gens | ID-SPN         |
|-------------|---------------|----------|----------------|
| NLTCS       | <b>-6</b>     | -6.11    | -6.02          |
| MSNBC       | -6.1          | -6.11    | <b>-6.04</b>   |
| KDDCup 2k   | -2.2          | -2.18    | <b>-2.13</b>   |
| Plants      | <b>-11.99</b> | -12.98   | -12.54         |
| Audio       | -41.02        | -40.5    | <b>-39.79</b>  |
| Jester      | <b>-41.11</b> | -75.99   | -52.86         |
| Netflix     | -58.02        | -57.33   | <b>-56.36</b>  |
| Accidents   | <b>-24.87</b> | -30.04   | -26.98         |
| Retail      | <b>-10.6</b>  | -11.04   | -10.85         |
| Pumsb-star  | -23.7         | -24.78   | <b>-22.4</b>   |
| DNA         | <b>-80.07</b> | -82.52   | -81.21         |
| Kosarak     | <b>-10.57</b> | -10.99   | -10.6          |
| MSWeb       | <b>-9.22</b>  | -10.25   | -9.73          |
| Book        | <b>-30.18</b> | -35.89   | -34.14         |
| EachMovie   | -52.47        | -52.49   | <b>-51.51</b>  |
| WebKB       | -153.5        | -158.2   | <b>-151.84</b> |
| Reuters-52  | <b>-82.1</b>  | -85.07   | -83.35         |
| 20 Newsgrp. | -152.39       | -155.93  | <b>-151.47</b> |
| BBC         | -251          | -250.69  | <b>-248.93</b> |
| Ad          | <b>-17.82</b> | -19.73   | -19            |

Table 6 shows the results for DSPN-SVD, SPN-SVD, SPN-Gens and ID-SPN. Apart from boosting algorithms, DSPN-SVD achieves higher accuracy than other algorithms on USPS, including C4.5 as reported in Demiriz et al. [2002]. As per MNIST, DSPN-SVD also achieves the highest accuracy for an SPN, and 2.2% less than the current overall state-of-the-art accuracy on MNIST (reported as 99.79% by Wan et al. [2013] and 99.77% by Ciresan et al. [2012]). The flexibility of extracting features and building a discriminative SPN tailored for the respective dataset makes DSPN-SVD superior to SPN-SVD, as well as SPN-Gens and ID-SPN on both USPS and MNIST.

Table 6: Classification of Handwritten Digits.

| Dataset | DSPN-SVD | SPN-SVD | SPN-Gens | ID-SPN |
|---------|----------|---------|----------|--------|
| USPS    | 92.4%    | 90.2%   | 79%      | 77.1%  |
| MNIST   | 97.6%    | 85%     | 81.8%    | 83.4%  |



Figure 5: Face Image Completions. The top row shows the original images; the bottom row shows images with the left half inferred using SPN-SVD.

## 5 CONCLUSION

State-of-the-art results when performing learning and inference on image datasets and digit classification indicate that the proposed SPN structure learning algorithms are effective.

Some important advantages of SPN-SVD over previously developed approaches are that it: (i) does not depend on local data splittings and instead globally splits the data based on rank-1 submatrix extraction; (ii) is based on correlations, which are easier to estimate than independences; and (iii) achieves considerable speedups by detecting large approximate rank-1 submatrices and avoiding redundant computations.

Interesting directions for future research include extending the discriminative setting to regression or structured-output learning by plugging more sophisticated kernels into the HSIC step, and enabling the SPN to model features optimised for different labels.

## References

- F R Bach. Consistency of the Group Lasso and Multiple Kernel Learning. *JMLR*, 2008.
- E. Barshan, A. Ghodsi, Z. Azimifar, and M. Jahromi. Su-

- pervised principal component analysis: visualization, classification and regression on subspaces and submanifolds. *In Pattern Recognition*, 44:1357–1371, 2011.
- M. Biggs, A. Ghodsi, and S. Vavasis. Nonnegative matrix factorization via rank-one downdate. *In International Conference on Machine Learning (ICML)*, 25, 2008a.
- M. Biggs, A. Ghodsi, and S. Vavasis. Nonnegative matrix factorization via rank-one downdate. *In <http://www.arxiv.org/abs/0805.0120>*, 2008b.
- D. M. Chickering. The winmine toolkit. *Microsoft, Redmond, WA MSR-TR-2002-103*, 2002.
- D. Ciresan, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3642–3649, 2012.
- A. Demiriz, K. P. Bennett, and J. Shawe-Taylor. Linear programming boosting via column generation. *In Machine Learning*, 46:225–254, 2002.
- A. Dennis and D. Ventura. Learning the architecture of sum-product networks using clustering on variables. *In NIPS*, 25, 2012.
- L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples. *In proceedings of CVPR Workshop on Generative Model-Based Vision*, 2004.
- K. Fukumizu, F. R. Bach, and M. I. Jordan. Dimensionality reduction for supervised learning with reproducing kernel Hilbert spaces. *JMLR*, 5:73–99, 2004.
- R. Gens and P. Domingos. Discriminative learning of sum-product networks. *In NIPS*, 25, 2012.
- R. Gens and P. Domingos. Learning the structure of sum-product networks. *In International Conference on Machine Learning (ICML)*, 30, 2013.
- A. Gretton, O. Bousquet, A. Smola, and B. Scholkopf. Statistical dependence with Hilbert-Schmidt norms. *In Algorithmic Learning Theory (ALT)*, 3734:63–77, 2005.
- J. J. Hull. A database for handwritten text recognition research. *In IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):550–554, 1994.
- Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *In Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- S.-W. Lee, H. Min-Oh, and Z. Byoung-Tak. Online incremental structure learning of sum-product networks. *In Neural Information Processing*, 2013.
- A. Nath and P. Domingos. Learning tractable statistical relational models. *In Workshop on Learning Tractable Probabilistic Models (LTPM)*, 2014.
- R. Peharz, B. C. Geiger, and F. Pernkopf. Greedy pairwise learning of sum-product networks. *In Machine Learning and Knowledge Discovery in Databases*, 8189:612–627, 2013.
- R. Peharz, R. Gens, and P. Domingos. Learning selective sum-product networks. *In Workshop on Learning Tractable Probabilistic Models (LTPM)*, 2014.
- H. Poon and P. Domingos. Sum-product networks: A new deep architecture. *In UAI*, 27, 2011.
- A. Rooshenas and D. Lowd. Learning sum-product networks with direct and indirect variable interactions. *In International Conference on Machine Learning (ICML)*, 31, 2014.
- R. Salakhutdinov and G. Hinton. Deep Boltzmann machines. *In AISTATS*, pages 448–455, 2009.
- Le Song, Alex J. Smola, Arthur Gretton, Karsten Borgwardt, and Justin Bedo. Supervised Feature Selection via Dependence Estimation. *In ICML*, 2007.
- L. Wan, M. Zeiler, S. Zhang, Y. LeCun, and R. Fergus. Regularization of neural networks using DropConnect. *In International Conference on Machine Learning (ICML)*, 30:1058–1066, 2013.

---

# Robust reconstruction of causal graphical models based on conditional 2-point and 3-point information

---

S everine Affeldt, Herv e Isambert

Institut Curie, Research Center, CNRS, UMR168, 26 rue d’Ulm, 75005, Paris France;  
and Universit e Pierre et Marie Curie, 4 Place Jussieu, 75005, Paris, France  
herve.isambert@curie.fr

## Abstract

We report a novel network reconstruction method, which combines constraint-based and Bayesian frameworks to reliably reconstruct graphical models despite inherent sampling noise in finite observational datasets. The approach is based on an information theory result tracing back the existence of colliders in graphical models to negative conditional 3-point information between observed variables. In turn, this provides a confident assessment of structural independencies in causal graphs, based on the ranking of their most likely contributing nodes with (significantly) positive conditional 3-point information. Starting from a complete undirected graph, dispensable edges are progressively pruned by iteratively “taking off” the most likely positive conditional 3-point information from the 2-point (mutual) information between each pair of nodes. The resulting network skeleton is then partially directed by orienting and propagating edge directions, based on the sign and magnitude of the conditional 3-point information of unshielded triples. This “3off2” network reconstruction approach is shown to outperform constraint-based, search-and-score and earlier hybrid methods on a range of benchmark networks.

## 1 INTRODUCTION

The prospect of learning the direction of causal dependencies from mere correlations in observational data has long defied practical implementations (Reichenbach, 1956). The fact that causal relationships can, to some extent, be inferred from nontemporal statistical data is now known to hinge on the unique statistical imprint of colliders in causal graphical models, provided that certain assumptions are made about the underlying process of data generation, such as its faithfulness to a tree structure (Rebane

and Pearl, 1988) or a directed acyclic graph model (Spirtes, Glymour, and Scheines, 2000; Pearl, 2009).

These early findings led to the developments of two types of network reconstruction approaches; on the one hand, search and score methods (Cooper and Herskovits, 1992; Heckerman, Geiger, and Chickering, 1995; Chickering, 2002) need heuristic strategies, such as hill-climbing algorithms, to sample network space, on the other hand, constraint-based methods, such as the PC (Spirtes and Glymour, 1991) and IC (Pearl and Verma, 1991) algorithms, rely on the identification of structural independencies, that correspond to edges to be removed from the underlying network (Spirtes, Glymour, and Scheines, 2000; Pearl, 2009). Yet, early errors in removing edges from the complete graph often lead to the accumulation of compensatory errors later on in the pruning process. Hence, despite recent, more stable implementations intending to overcome order-dependency in the pruning process (Colombo and Maathuis, 2014), constraint-based methods are not robust to sampling noise in finite datasets.

In this paper, we present a more robust constrained-based method and corresponding 3off2 algorithm. It is directly inspired by the PC and IC algorithms but relies on a quantitative information theoretic framework to reliably uncover conditional independencies in finite datasets and subsequently orient and propagate edge directions between connected variables.

## 2 RESULTS

### 2.1 UNCOVERING CAUSALITY FROM A STABLE / FAITHFUL DISTRIBUTION

Consider a network  $\mathcal{G} = (V, E)$  and a *stable* (or *faithful*) distribution  $P(\mathbf{X})$  over  $V$ , implying that each structural independency (*i.e.* missing edge  $XY$  in  $\mathcal{G}$ ) corresponds to a vanishing conditional 2-point (mutual) information and reciprocally as,

$$(X \perp\!\!\!\perp Y | \{U_i\})_{\mathcal{G}} \iff (X \perp\!\!\!\perp Y | \{U_i\})_P \quad (1)$$

$$\iff I(X; Y | \{U_i\}) = 0 \quad (2)$$

Eq. 1 assumes, in particular, that  $P(\mathbf{X})$  is a theoretical distribution, defined by a formal expression of its variables  $\mathbf{X} = \{X, Y, U_1, U_2, \dots\}$ . Note, however, that no such expression is known *a priori*, in general, and  $P(\mathbf{X})$  must typically be *estimated* from the available data. In principle, an infinite amount of data would be necessary to infer an ‘exact’ *stable* distribution  $P(\mathbf{X})$  consistent with Eq. 1. In the following, we will first assume that such an infinite amount of data is available and distributed as a stable  $P(\mathbf{X})$  to establish how causality can be inferred statistically from conditional 2-point and 3-point information. We will then consider the more realistic situation for which  $P(\mathbf{X})$  is not known exactly and must be estimated from a finite amount of data.

Let us first recall the *generic decomposition* of a conditional 2-point (or mutual) information  $I(X; Y | \{U_i\})$  by the introduction of a third node  $Z$  and the conditional 3-point information  $I(X; Y; Z | \{U_i\})$ ,

$$I(X; Y | \{U_i\}) = I(X; Y; Z | \{U_i\}) + I(X; Y | \{U_i\}, Z) \quad (3)$$

This relation can be taken as the definition of conditional 3-point information  $I(X; Y; Z | \{U_i\})$  which is in fact symmetric in  $X, Y$  and  $Z$ ,

$$\begin{aligned} I(X; Y; Z | \{U_i\}) &= I(X; Y | \{U_i\}) - I(X; Y | \{U_i\}, Z) \\ &= I(X; Z | \{U_i\}) - I(X; Z | \{U_i\}, Y) \\ &= I(Y; Z | \{U_i\}) - I(Y; Z | \{U_i\}, X) \end{aligned}$$

Note that Eq. 3 is always valid, regardless of any assumption on the underlying graphical model and of the amount of data available to estimate conditional 2-point and 3-point information terms. Eq. 3 will be used to prove the following lemmas and propositions, which trace back the origin of necessary causal relationships in a graphical model to the existence of a negative conditional 3-point information between *three* variables  $\{X, Y, Z\}$ ,  $I(X; Y; Z | \{U_i\}) < 0$ , where  $\{U_i\}$  accounts for a structural independency between two of them, *e.g.*  $I(X; Y | \{U_i\}) = 0$  (see Theorem 4).

**Lemma 1.** *Given a stable distribution  $P(\mathbf{X})$  on  $V$ ,  $\forall X, Y \in V$  not adjacent in  $\mathcal{G}$ ,  $\exists \{U_i\} \subseteq V_{\setminus \{X, Y\}}$  s.t.  $I(X; Y | \{U_i\}) = 0$  and  $\forall Z \neq X, Y, \{U_i\}$ ,  $I(X; Y; Z | \{U_i\}) \leq 0$ .*

**Proof.** If  $X, Y \in V$  are not adjacent in  $\mathcal{G}$ , this corresponds to a structural independency, *i.e.*  $\exists \{U_i\} \subseteq V_{\setminus \{X, Y\}}$  s.t.  $I(X; Y | \{U_i\}) = 0$ . Then  $\forall Z \neq X, Y, \{U_i\}$  Eq. 3 implies  $I(X; Y; Z | \{U_i\}) = -I(X; Y | \{U_i\}, Z) \leq 0$ , as conditional mutual information is always positive.  $\square$

**Corollary 2 (3-point contribution).**  $\forall X, Y, Z \in V$  and  $\forall \{U_i\} \subseteq V_{\setminus \{X, Y, Z\}}$  s.t.  $I(X; Y; Z | \{U_i\}) > 0$ , then  $I(X; Y | \{U_i\}) > 0$  (as well as  $I(X; Z | \{U_i\}) > 0$  and  $I(Y; Z | \{U_i\}) > 0$  by symmetry of  $I(X; Y; Z | \{U_i\})$ ).

Corollary 2, which is a direct consequence of Eq. 3 and the positivity of mutual information, will be the ba-

sis of the 3off2 causal network reconstruction algorithm, which iteratively ‘takes off’ 3-point information from 2-point information, as  $I(X; Y | \{U_i\}) - I(X; Y; Z | \{U_i\}) = I(X; Y | \{U_i\}, Z)$ , and update  $\{U_i\} \leftarrow \{U_i\} + Z$  as long as there remains some  $Z \in V$  with (significantly) positive conditional 3-point information  $I(X; Y; Z | \{U_i\}) > 0$ .

**Lemma 3 (vanishing conditional 2-point and 3-point information in undirected networks).** *If  $\mathcal{G}$  is an undirected (Markov) network,  $\forall X, Y \in V$  and  $\forall \{U_i\} \subseteq V_{\setminus \{X, Y\}}$  s.t.  $I(X; Y | \{U_i\}) = 0$ , then  $\forall Z \neq X, Y, \{U_i\}$ ,  $I(X; Y; Z | \{U_i\}) = 0$ .*

**Proof.** If  $\mathcal{G}$  is a Markov network,  $\forall X, Y \in V$  and  $\forall \{U_i\} \subseteq V_{\setminus \{X, Y\}}$  s.t.  $I(X; Y | \{U_i\}) = 0$ , then  $\forall Z \neq X, Y, \{U_i\}$ ,  $I(X; Y | \{U_i\}, Z) = 0$  as conditioning observation cannot induce correlations in Markov networks (Koller and Friedman, 2009). This implies that  $I(X; Y; Z | \{U_i\}) = 0$  through Eq. 3.  $\square$

Note, however, that the converse of Lemma 3 is not true. Namely, (partially) directed networks can also have vanishing conditional 3-point information associated to all their structural independencies. In particular, tree-like bayesian networks without colliders (*i.e.* without v-structures,  $X \rightarrow Z \leftarrow Y$ ) present only vanishing 3-point information associated to their structural independencies, *i.e.*  $I(X; Y; Z | \{U_i\}) = 0$ ,  $\forall X, Y, Z, \{U_i\} \in V$  s.t.  $I(X; Y | \{U_i\}) = 0$ . However, such a directed network must be Markov equivalent to an undirected network corresponding to the same structural independencies but lacking any trace of causal relationships (*i.e.* no directed edges). The probability distributions faithful to such directed networks do not contain evidence of obligate causality; *i.e.* no directed edges can be unambiguously oriented.

The following Theorem 4 establishes the existence of negative conditional 3-point information as statistical evidence of obligate causality in graphical models. For the purpose of generality in this section, we do not exclude the possibility that unobserved ‘latent’ variables might mediate the causal relationships among observed variables. However, this requires dissociating the labelling of the two endpoints of each edges. Let us first introduce three different endpoint marks associated to such edges in mixed graphs: they are the tail ( $-$ ), the head ( $>$ ) and the unspecified ( $\circ$ ) endpoint marks. In addition, we will use the asterisk symbol ( $*$ ) as a wild card denoting any of the three marks.

**Theorem 4 (negative conditional 3-point information as statistical evidence of causality).** *If  $\exists X, Y, Z \in V$  and  $\{U_i\} \subseteq V_{\setminus \{X, Y, Z\}}$  s.t.  $I(X; Y | \{U_i\}) = 0$  and  $I(X; Y; Z | \{U_i\}) < 0$  then,  $\mathcal{G}$  is (partially) directed, *i.e.* some variables in  $\mathcal{G}$  are causally linked, either directly or indirectly through other variables, including possibly unknown, ‘latent’ variables unobserved in  $\mathcal{G}$ .*

**Proof.** Theorem 4 is the contrapositive of Lemma 3, with the additional use of Lemma 1.  $\square$

**Proposition 5 (origin of causality at unshielded triples with negative conditional 3-point information).**

For all unshielded triple,  $X * - \circ Z \circ - * Y$ ,  $\exists \{U_i\} \subseteq V_{\setminus \{X, Y\}}$  s.t.  $I(X; Y | \{U_i\}) = 0$ , if  $Z \notin \{U_i\}$  then  $I(X; Y; Z | \{U_i\}) < 0$  and the unshielded triple should be oriented as  $X * \rightarrow Z \leftarrow * Y$ .

**Proof.** If  $I(X; Y | \{U_i\}) = 0$  with  $Z \notin \{U_i\}$ , the unshielded triple has to be a collider and  $I(X; Y | \{U_i\}, Z) > 0$ , by faithfulness, hence,  $I(X; Y; Z | \{U_i\}) < 0$  by Eq. 3.  $\square$

Hence, the origin of causality manifests itself in the form of colliders or v-structures in graphical models which reveal ‘genuine’ causations ( $X \rightarrow Z$  or  $Y \rightarrow Z$ ) or, alternatively, ‘possible’ causations ( $X \circ \rightarrow Z$  or  $Y \circ \rightarrow Z$ ), provided that the corresponding correlations are not due to unobserved ‘latent’ variables  $L$  or  $L'$  as,  $X \leftarrow - L - \rightarrow Z$  or  $Y \leftarrow - L' - \rightarrow Z$ .

Following the rationale of constraint-based approaches, it is then possible to ‘propagate’ further the orientations downstream of colliders, through positive (conditional) 3-point information, if one assumes that the underlying distribution  $P(\mathbf{X})$  is faithful to an *ancestral graph*  $\mathcal{G}$  on  $V$ . An *ancestral graph* is a mixed graph, that is, with three types of edges, undirected ( $-$ ), directed ( $\leftarrow$  or  $\rightarrow$ ) or bidirectional ( $\leftrightarrow$ ), but with *i.*) no directed cycle, *ii.*) no almost directed cycle (including one bidirectional edge) and *iii.*) no undirected edge with incoming arrowhead (such as  $X * \rightarrow Z - Y$ ). In particular, Directed Acyclic Graphs (DAG) are subclasses of ancestral graphs (*i.e.* without undirected nor bidirectional edges).

**Proposition 6 (‘propagation’ of causality at unshielded triples with positive conditional 3-pt information).**

Given a distribution  $P(\mathbf{X})$  faithful to an ancestral graph  $\mathcal{G}$  on  $V$ , for all unshielded triple with already one converging orientation,  $X * \rightarrow Z \circ - * Y$ ,  $\exists \{U_i\} \subseteq V_{\setminus \{X, Y\}}$  s.t.  $I(X; Y | \{U_i\}) = 0$ , if  $Z \in \{U_i\}$  then  $I(X; Y; Z | \{U_i\} \setminus Z) > 0$  and the first orientation should be ‘propagated’ to the second edge as  $X * \rightarrow Z \rightarrow Y$ .

**Proof.** If  $I(X; Y | \{U_i\}) = 0$  with  $Z \in \{U_i\}$ , the unshielded triple cannot be a collider and, since  $\mathcal{G}$  is assumed to be an ancestral graph, the edge  $Z - Y$  cannot be an undirected edge either. Hence, it has to be a directed edge,  $Z \rightarrow Y$  and  $I(X; Y; Z | \{U_i\} \setminus Z) > 0$  by faithfulness and Eq. 3.  $\square$

Note that the propagation rule of Proposition 6 can be applied iteratively to successive unshielded triples corresponding to positive conditional 3-point information. Yet, all arrowhead orientations can be ultimately traced back to a negative conditional 3-point information, Theorem 4 and

Proposition 5.

**2.2 ROBUST RECONSTRUCTION OF CAUSAL GRAPHS FROM FINITE DATASETS**

We now turn to the more practically relevant situation of finite datasets consisting of  $N$  independent data points. The associated sampling noise will intrinsically limit the accuracy of causal network reconstruction. In particular, conditional independencies cannot be exactly achieved ( $I(X; Y | \{U_i\}) = 0$ ) but can be reliably established using statistical criteria that depend on the number of data points  $N$ .

Given  $N$  independent datapoints from the available data  $\mathcal{D}$ , let us introduce the maximum likelihood,  $\mathcal{L}_{\mathcal{D}|\mathcal{G}}$ , that they might have been generated by the graphical model  $\mathcal{G}$  (Sanov, 1957),

$$\mathcal{L}_{\mathcal{D}|\mathcal{G}} = \frac{e^{-NH(\mathcal{G}, \mathcal{D})}}{Z(\mathcal{G}, \mathcal{D})} = \frac{e^{N \sum_{\{x_i\}} p(\{x_i\}) \log(q(\{x_i\}))}}{Z(\mathcal{G}, \mathcal{D})} \quad (4)$$

where  $H(\mathcal{G}, \mathcal{D}) = -\sum_{\{x_i\}} p(\{x_i\}) \log(q(\{x_i\}))$  is the cross entropy between the ‘true’ probability distribution  $p(\{x_i\})$  of the data  $\mathcal{D}$  and the theoretical probability distribution  $q(\{x_i\})$  of the model  $\mathcal{G}$  and  $Z(\mathcal{G}, \mathcal{D})$  is a data- and model-dependent factor ensuring proper normalization condition. The structural constraints of the model  $\mathcal{G}$  can be included *a priori* in the factorization form of the theoretical probability distribution,  $q(\{x_i\})$ . In particular, if we assume a Bayesian network as underlying graphical model,  $q(\{x_i\})$  factorizes as  $q(\{x_i\}) = \prod_i p(x_i | \text{pa}_{x_i})$ , where  $\{\text{pa}_{x_i}\}$  denote the values of the parents of node  $X_i$ ,  $\{\text{Pa}_{X_i}\}$ , and leads to the following maximum likelihood expression,

$$\mathcal{L}_{\mathcal{D}|\mathcal{G}} = \frac{e^{-N \sum_i H(X_i | \{\text{Pa}_{X_i}\})}}{Z(\mathcal{G}, \mathcal{D})} \quad (5)$$

The model  $\mathcal{G}$  can then be compared to the alternative model  $\mathcal{G}_{\setminus X \rightarrow Y}$  with one additional missing edge  $X \rightarrow Y$  using the maximum likelihood ratio,

$$\frac{\mathcal{L}_{\mathcal{D}|\mathcal{G}_{\setminus X \rightarrow Y}}}{\mathcal{L}_{\mathcal{D}|\mathcal{G}}} = e^{-NI(X; Y | \{\text{Pa}_Y \setminus X\})} \frac{Z(\mathcal{G}, \mathcal{D})}{Z(\mathcal{G}_{\setminus X \rightarrow Y}, \mathcal{D})} \quad (6)$$

where  $I(X; Y | \{\text{Pa}_Y \setminus X\}) = H(Y | \{\text{Pa}_Y \setminus X\}) - H(Y | \{\text{Pa}_Y\})$ . However, Eq. 6 cannot be used as such to learn the underlying graphical model, as it assumes that the order between the nodes and their parents is already known (see however (de Campos, 2006)). Yet, following the rationale of constraint-based approaches, Eq. 6 can be reformulated by replacing the parent nodes with an unknown separation set  $\{U_i\}$  to be learnt simultaneously with the missing edge candidate  $XY$ ,

$$\frac{\mathcal{L}_{\mathcal{G}_{\setminus XY | \{U_i\}}}}{\mathcal{L}_{\mathcal{G}}} = e^{-NI(X; Y | \{U_i\}) + k_{X; Y | \{U_i\}}} \quad (7)$$

$$k_{X; Y | \{U_i\}} = \log \left( Z(\mathcal{G}, \mathcal{D}) / Z(\mathcal{G}_{\setminus XY | \{U_i\}}, \mathcal{D}) \right)$$

where the factor  $k_{X;Y|\{U_i\}} > 0$  tends to limit the complexity of the models by favoring fewer edges. Namely, the condition,  $I(X;Y|\{U_i\}) < k_{X;Y|\{U_i\}}/N$ , implies that simpler models compatible with the structural independency,  $X \perp\!\!\!\perp Y|\{U_i\}$ , are more likely than model  $\mathcal{G}$ , given the finite available dataset. This replaces the ‘perfect’ conditional independency condition,  $I(X;Y|\{U_i\}) = 0$ , valid in the limit of an infinite dataset,  $N \rightarrow \infty$ . A common complexity criteria in model selection is the Bayesian Information Criteria (BIC) or Minimal Description Length (MDL) criteria (Rissanen, 1978; Hansen and Yu, 2001),

$$k_{X;Y|\{U_i\}}^{\text{MDL}} = \frac{1}{2}(r_x - 1)(r_y - 1) \prod_i r_{u_i} \log N \quad (8)$$

where  $r_x, r_y$  and  $r_{u_i}$  are the number of levels of the corresponding variables. The MDL complexity, Eq. 8, is simply related to the normalisation constant reached in the asymptotic limit of a large dataset  $N \rightarrow \infty$  (Laplace approximation). However, this limit distribution is only reached for very large datasets in practice. Alternatively, the normalisation of the maximum likelihood can also be done over all possible datasets including the same number of data points to yield a (universal) Normalized Maximum Likelihood (NML) criteria (Shtarkov, 1987; Rissanen and Tabus, 2005) and its decomposable (Kontkanen and Myllymäki, 2007; Roos et al., 2008) and  $XY$ -symmetric version,  $k_{X;Y|\{U_i\}}^{\text{NML}}$ , defined in the Supplementary Methods.

Then, instead of exploring the combinatorics of sepset composition  $\{U_i\}$  for each missing edge candidate  $XY$  as in traditional constraint-based approaches, we propose that Eq. 7 can be used to iteratively extend a *likely* sepset using the maximum likelihood ratios between two successive sepset candidates, *i.e.* between the already ascertained  $\{U_i\}$  and the possible extended  $\{U_i\} + Z$ , as,

$$\frac{\mathcal{L}_{\mathcal{D}|\mathcal{G}_{\setminus XY|\{U_i\},Z}}}{\mathcal{L}_{\mathcal{D}|\mathcal{G}_{\setminus XY|\{U_i\}}}} = e^{NI(X;Y;Z|\{U_i\}) + k_{X;Y;Z|\{U_i\}}} \quad (9)$$

using Eq. 3 for  $I(X;Y;Z|\{U_i\})$  and introducing a similar 3-point complexity conditioned on  $\{U_i\}$  as,

$$k_{X;Y;Z|\{U_i\}} = k_{X;Y|\{U_i\},Z} - k_{X;Y|\{U_i\}} \quad (10)$$

where  $k_{X;Y;Z|\{U_i\}} \geq 0$ , unlike 3-point information,  $I(X;Y;Z|\{U_i\})$  which can be positive or negative.

Introducing also the shifted 2-point and 3-point information for finite datasets as,

$$\begin{aligned} I'(X;Y|\{U_i\}) &= I(X;Y|\{U_i\}) - \frac{k_{X;Y|\{U_i\}}}{N} \\ I'(X;Y;Z|\{U_i\}) &= I(X;Y;Z|\{U_i\}) + \frac{k_{X;Y;Z|\{U_i\}}}{N} \end{aligned}$$

leads to maximum likelihood ratios equivalent to Eq. 7 and

Eq. 9,

$$\frac{\mathcal{L}_{\mathcal{D}|\mathcal{G}_{\setminus XY|\{U_i\}}}}{\mathcal{L}_{\mathcal{D}|\mathcal{G}}} = e^{-NI'(X;Y|\{U_i\})} \quad (11)$$

$$\frac{\mathcal{L}_{\mathcal{D}|\mathcal{G}_{\setminus XY|\{U_i\},Z}}}{\mathcal{L}_{\mathcal{D}|\mathcal{G}_{\setminus XY|\{U_i\}}}} = e^{NI'(X;Y;Z|\{U_i\})} \quad (12)$$

As will become apparent in the following discussion, learning, iteratively, the most likely edge to be removed  $XY$  and its corresponding separation set  $\{U_i\}$  will imply to simultaneously minimize 2-point information (Eq. 11) while maximizing 3-point information (Eq. 12).

We start the discussion with 3-point information, Eq. 12. The sign and magnitude of shifted conditional 3-point information  $I'(X;Y;Z|\{U_i\})$  determine the probability that  $Z$  should be included in or excluded from the sepset candidate  $\{U_i\}$ ,

- If  $I'(X;Y;Z|\{U_i\}) > 0$ ,  $Z$  is more likely to be included in  $\{U_i\}$  with probability,

$$\begin{aligned} P_{\text{nv}}(X;Y;Z|\{U_i\}) &= \frac{\mathcal{L}_{\mathcal{D}|\mathcal{G}_{\setminus XY|\{U_i\},Z}}}{\mathcal{L}_{\mathcal{D}|\mathcal{G}_{\setminus XY|\{U_i\}}} + \mathcal{L}_{\mathcal{D}|\mathcal{G}_{\setminus XY|\{U_i\},Z}}} \\ &= \frac{1}{1 + e^{-NI'(X;Y;Z|\{U_i\})}} \quad (13) \end{aligned}$$

- If  $I'(X;Y;Z|\{U_i\}) < 0$ ,  $Z$  is more likely to be excluded from  $\{U_i\}$ , suggesting obligatory causal relationships in the form of a v-structure or collider between  $X, Y, Z$  with probability,

$$\begin{aligned} P_{\text{v}}(X;Y;Z|\{U_i\}) &= 1 - P_{\text{nv}}(X;Y;Z|\{U_i\}) \\ &= \frac{1}{1 + e^{NI'(X;Y;Z|\{U_i\})}} \quad (14) \end{aligned}$$

But, in the case  $I'(X;Y;Z|\{U_i\}) > 0$ , Eq. 12 can also be interpreted as quantifying the likelihood increase that the edge  $XY$  should be removed from the model by extending the candidate sepset from  $\{U_i\}$  to  $\{U_i\} + Z$ , *i.e.*  $\mathcal{L}_{\mathcal{D}|\mathcal{G}_{\setminus XY|\{U_i\},Z}} = \mathcal{L}_{\mathcal{D}|\mathcal{G}_{\setminus XY|\{U_i\}}} \times \exp(NI'(X;Y;Z|\{U_i\}))$ , with  $\exp(NI'(X;Y;Z|\{U_i\})) > 1$ . Yet, as the 3-point information,  $I'(X;Y;Z|\{U_i\})$ , is actually symmetric with respect to the variables,  $X, Y$  and  $Z$ , the factor  $\exp(NI'(X;Y;Z|\{U_i\})) > 1$  provides in fact the same likelihood increase for the removal of the three edges  $XY, XZ$  and  $ZY$ , conditioned on the same initial set of nodes  $\{U_i\}$ , namely,

$$\begin{aligned} \frac{\mathcal{L}_{\mathcal{D}|\mathcal{G}_{\setminus XY|\{U_i\},Z}}}{\mathcal{L}_{\mathcal{D}|\mathcal{G}_{\setminus XY|\{U_i\}}}} &= \frac{\mathcal{L}_{\mathcal{D}|\mathcal{G}_{\setminus XZ|\{U_i\},Y}}}{\mathcal{L}_{\mathcal{D}|\mathcal{G}_{\setminus XZ|\{U_i\}}}} = \frac{\mathcal{L}_{\mathcal{D}|\mathcal{G}_{\setminus ZY|\{U_i\},X}}}{\mathcal{L}_{\mathcal{D}|\mathcal{G}_{\setminus ZY|\{U_i\}}}} \\ &= e^{NI'(X;Y;Z|\{U_i\})} \end{aligned}$$

However, despite this symmetry of 3-point information,  $I'(X; Y; Z|\{U_i\})$ , the likelihoods that the edges  $XY$ ,  $XZ$  and  $ZY$  should be removed are not the same, as they depend on different 2-point information,  $I'(X; Y|\{U_i\})$ ,  $I'(X; Z|\{U_i\})$  and  $I'(Z; Y|\{U_i\})$ , Eq. 11. In particular, the likelihood ratio between the removals of the alternative edges  $XY$  and  $XZ$  is given by,

$$\frac{\mathcal{L}_{\mathcal{D}|G_{\setminus XY|\{U_i\}, Z}}}{\mathcal{L}_{\mathcal{D}|G_{\setminus XZ|\{U_i\}, Y}}} = \frac{\mathcal{L}_{\mathcal{D}|G_{\setminus XY|\{U_i\}}}}{\mathcal{L}_{\mathcal{D}|G_{\setminus XZ|\{U_i\}}}} = \frac{e^{-NI'(X; Y|\{U_i\})}}{e^{-NI'(X; Z|\{U_i\})}} \quad (15)$$

and similarly between edges  $XY$  and  $ZY$ .

Hence, for  $XY$  to be the most likely edge to be removed conditioned on the sepset  $\{U_i\} + Z$ , not only  $Z$  should contribute through  $I'(X; Y; Z|\{U_i\}) > 0$  with probability  $P_{nv}(X; Y; Z|\{U_i\})$  (Eq. 13), but  $XY$  must also correspond to the ‘weakest’ edge of  $XY$ ,  $XZ$  and  $ZY$  conditioned on  $\{U_i\}$ , as given by the lowest conditioned 2-point information, Eq. 15. Note that removing the edge  $XY$  with the lowest conditional 2-point information is consistent, as expected, with the Data Processing Inequality,  $I(X; Y|\{U_i\}) \leq \min(I(X; Z|\{U_i\}), I(Z; Y|\{U_i\}))$ , in the limit of large datasets. However, quite frequently,  $XZ$  or  $ZY$  might also have low conditional 2-point information, so that the edge removal associated with the symmetric contribution  $I(X; Y; Z|\{U_i\})$  will only be consistent with the Data Processing Inequality (DPI) with probability,

$$\begin{aligned} P_{dpi}(XY; Z|\{U_i\}) &= \\ &= \frac{\mathcal{L}_{\mathcal{D}|G_{\setminus XY|\{U_i\}}}}{\mathcal{L}_{\mathcal{D}|G_{\setminus XY|\{U_i\}}} + \mathcal{L}_{\mathcal{D}|G_{\setminus XZ|\{U_i\}}} + \mathcal{L}_{\mathcal{D}|G_{\setminus ZY|\{U_i\}}}} \\ &= \frac{1}{1 + \frac{e^{-NI'(X; Z|\{U_i\})}}{e^{-NI'(X; Y|\{U_i\})}} + \frac{e^{-NI'(Z; Y|\{U_i\})}}{e^{-NI'(X; Y|\{U_i\})}}} \quad (16) \end{aligned}$$

In practice, taking into account this DPI-consistency probability  $P_{dpi}(XY; Z|\{U_i\})$ , as detailed below, significantly improves the results obtained by relying solely on the ‘non-structure’ probability  $P_{nv}(X; Y; Z|\{U_i\})$ . Conversely, the DPI-consistency probability  $P_{dpi}(XY; Z|\{U_i\})$  is not sufficient on its own to uncover causal relationships between variables, which require to compute 3-point information  $I(X; Y; Z|\{U_i\})$  and the probability  $P_{nv}(X; Y; Z|\{U_i\})$  (see Proposition 7 and Proposition 8, below).

To optimize the likelihood that the edge  $XY$  can be accounted for by the additional contribution of  $Z$  conditioned on previously selected  $\{U_i\}$ , we propose to combine the maximum of 3-point information (Eq. 13) and the minimum of 2-point information (Eq. 16) by defining the score  $S_{lb}(Z; XY|\{U_i\})$  as the lower bound of  $P_{nv}(X; Y; Z|\{U_i\})$  and  $P_{dpi}(XY; Z|\{U_i\})$ , since both conditions need to be fulfilled to warrant that edge  $XY$  is likely to be absent from

the model  $\mathcal{G}$ ,

$$\begin{aligned} S_{lb}(Z; XY|\{U_i\}) &= \\ &= \min \left[ P_{nv}(X; Y; Z|\{U_i\}), P_{dpi}(XY; Z|\{U_i\}) \right] \end{aligned}$$

Hence, the pair of nodes  $XY$  with the most likely contribution from a third node  $Z$  and likely to be absent from the model can be ordered according to their rank  $R(XY; Z|\{U_i\})$  defined as,

$$R(XY; Z|\{U_i\}) = \max_Z (S_{lb}(Z; XY|\{U_i\})) \quad (17)$$

Then,  $Z$  can be iteratively added to the set of contributing nodes (*i.e.*  $\{U_i\} \leftarrow \{U_i\} + Z$ ) of the top edge  $XY = \operatorname{argmax}_{XY} R(XY; Z|\{U_i\})$  to progressively recover the most significant indirect contributions to all pairwise mutual information in a causal graph.

Implementing this local optimization scheme, the 3off2 algorithm eventually learns the network skeleton by collecting the nodes of the separation sets one-by-one, instead of exploring the full combinatorics of sepset composition without any likelihood guidance. Indeed, the 3off2 scheme amounts to identify  $\{U_i\}$  by ‘‘taking off’’ iteratively the ‘‘most likely’’ conditional 3-point information from each 2-point information as,

$$\begin{aligned} I(X; Y|\{U_i\}_n) &= I(X; Y) - I(X; Y; U_1) \\ &\quad - I(X; Y; U_2|U_1) - \dots \\ &\quad - I(X; Y; U_n|\{U_i\}_{n-1}) \end{aligned}$$

or equivalently between the shifted 2-point and 3-point information terms,

$$\begin{aligned} I'(X; Y|\{U_i\}_n) &= I'(X; Y) - I'(X; Y; U_1) \\ &\quad - I'(X; Y; U_2|U_1) - \dots \\ &\quad - I'(X; Y; U_n|\{U_i\}_{n-1}) \end{aligned}$$

This leads to the following Algorithm 1 for the reconstruction of the graph skeleton using the 3off2 scheme. Note, in particular, that the 3off2 scheme to reconstruct graph skeleton is solely based on identifying structural independencies, which can also be applied to graphical models for undirected Markov networks.

Then, given the skeleton obtained from Algorithm 1, Eqs. 13 and 14 lead to the following Proposition 7 and Proposition 8 for the orientation and propagation rules of unshielded triples, which are equivalent to Proposition 5 and Proposition 6 but for underlying DAG models (assuming no latent variables) and for finite datasets with the corresponding probabilities for the initiation/propagation of orientations.



---

**Algorithm 1:** 3off2 Skeleton Reconstruction

---

**In:** observational data of finite size  $N$ **Out:** skeleton of causal graph  $\mathcal{G}$ **Initiation**

Start with complete undirected graph

**forall** edges  $XY$  **do**  **if**  $I'(X; Y) < 0$  **then**     $XY$  **edge is** non-essential and **removed**    **separation set** of  $XY$ :  $\text{Sep}_{XY} = \emptyset$   **else**    find the **most contributing node**  $Z$  neighbor of  $X$   
    or  $Y$  and **compute 3off2 rank**,  $R(XY; Z|\emptyset)$   **end****end****Iteration****while**  $\exists XY$  edge with  $R(XY; Z|\{U_i\}) > 1/2$  **do**  **for** edge  $XY$  with highest rank  $R(XY; Z|\{U_i\})$  **do**    **expand contributing set**  $\{U_i\} \leftarrow \{U_i\} + Z$     **if**  $I'(X; Y|\{U_i\}) < 0$  **then**       $XY$  **edge is** non-essential and **removed**      **separation set** of  $XY$ :  $\text{Sep}_{XY} = \{U_i\}$     **else**      find **next most contributing node**  $Z$  neighbor  
      of  $X$  or  $Y$  and **compute new 3off2 rank**:  
       $R(XY; Z|\{U_i\})$     **end**    **sort the 3off2 rank list**  $R(XY; Z|\{U_i\})$   **end****end**

---

**Proposition 7 (Significantly negative conditional 3-point information as robust statistical evidence of causality in finite datasets).**

Assuming that the underlying graphical model is a DAG  $\mathcal{G}$  on  $V$ ,  $\forall X, Y, Z \in V$  and  $\forall \{U_i\} \subseteq V \setminus \{X, Y, Z\}$  s.t.  $I'(X; Y|\{U_i\}) < 0$  (i.e. no  $XY$  edge) and  $I'(X; Y; Z|\{U_i\}) < 0$  then,

- i. if  $X, Y, Z$  form an unshielded triple,  $X \circ\text{-}\circ Z \circ\text{-}\circ Y$ , then it should be oriented as  $X \rightarrow Z \leftarrow Y$ , with probabilities,

$$P_{X \rightarrow Z}^\circ = P_{Y \rightarrow Z}^\circ = \frac{1 + e^{NI'(X; Y; Z|\{U_i\})}}{1 + 3e^{NI'(X; Y; Z|\{U_i\})}}$$

- ii. similarly, if  $X, Y, Z$  form an unshielded triple,

with one already known converging arrow,  $X \rightarrow Z \circ\text{-}\circ Y$ , with probability  $P_{X \rightarrow Z} > P_{X \rightarrow Z}^\circ$ , then the second edge should be oriented to form a  $v$ -structure,  $X \rightarrow Z \leftarrow Y$ , with probability,

$$P_{Y \rightarrow Z} = P_{X \rightarrow Z} \left( \frac{1}{1 + e^{NI'(X; Y; Z|\{U_i\})}} - \frac{1}{2} \right) + \frac{1}{2}$$

**Proof.** The implications (i.) and (ii.) rely on Eq. 14 to estimate the probability that the two edges form a collider. We start proving (ii.) using the probability decomposition formula:

$$\begin{aligned} P_{Y \rightarrow Z} &= P_{X \rightarrow Z} \frac{P_{X \rightarrow Z \leftarrow Y}}{P_{X \rightarrow Z \leftarrow Y} + P_{X \rightarrow Z \rightarrow Y}} \\ &\quad + (1 - P_{X \rightarrow Z}) \frac{P_{X \leftarrow Z \leftarrow Y}}{P_{X \leftarrow Z \leftarrow Y} + P_{X \leftarrow Z \rightarrow Y}} \\ &= P_{X \rightarrow Z} \left( \frac{1}{1 + e^{NI'(X; Y; Z|\{U_i\})}} - \frac{1}{2} \right) + \frac{1}{2} \end{aligned}$$

which also leads to (i.) if one assumes  $P_{X \rightarrow Z} = P_{Y \rightarrow Z}$  by symmetry in absence of prior information on these orientations.  $\square$

Following the rationale of constraint-based approaches, it is then possible to ‘propagate’ further the orientations downstream of colliders, using Eq. 13 for positive (conditional) 3-point information. For simplicity and consistency, we only implement the propagation of orientation based on likelihood ratios, which can be quantified for finite datasets as proposed in the following Proposition 8. In particular, we do not extend the propagation rules (Meek, 1995) to enforce acyclic constraints that are necessary to have a complete reconstruction of the Markov equivalent class of the underlying DAG model.

**Proposition 8 (robust ‘propagation’ of causality at unshielded triples with significantly positive conditional 3-pt information).**

Assuming that the underlying graphical model is a DAG  $\mathcal{G}$  on  $V$ ,  $\forall X, Y, Z \in V$  and  $\forall \{U_i\} \subseteq V \setminus \{X, Y, Z\}$  s.t.  $I'(X; Y|\{U_i\}, Z) < 0$  (i.e. no  $XY$  edge) and  $I'(X; Y; Z|\{U_i\}) > 0$ , then if  $X, Y, Z$  form an unshielded triple with one already known converging orientation,  $X \rightarrow Z \circ\text{-}\ast Y$ , with probability  $P_{X \rightarrow Z} > 1/2$ , this orientation should be ‘propagated’ to the second edge as  $X \rightarrow Z \rightarrow Y$ , with probability,

$$P_{Z \rightarrow Y} = P_{X \rightarrow Z} \left( \frac{1}{1 + e^{-NI'(X; Y; Z|\{U_i\})}} - \frac{1}{2} \right) + \frac{1}{2}$$

**Proof.** This results is shown using the probability decom-

position formula,

$$\begin{aligned}
P_{Z \rightarrow Y} &= P_{X \rightarrow Z} \frac{P_{X \rightarrow Z \rightarrow Y}}{P_{X \rightarrow Z \leftarrow Y} + P_{X \rightarrow Z \rightarrow Y}} \\
&\quad + (1 - P_{X \rightarrow Z}) \frac{P_{X \leftarrow Z \rightarrow Y}}{P_{X \leftarrow Z \leftarrow Y} + P_{X \leftarrow Z \rightarrow Y}} \\
&= P_{X \rightarrow Z} \left( \frac{1}{1 + e^{-NI'(X;Y;Z|\{U_i\})}} - \frac{1}{2} \right) + \frac{1}{2}
\end{aligned}$$

□

Proposition 7 and Proposition 8 lead to the following Algorithm 2 for the orientation of unshielded triples of the graph skeleton obtained from Algorithm 1.

---

**Algorithm 2:** 3off2 Orientation / Propagation Step

---

**In:** Graph skeleton from Algorithm 1 and corresponding conditional 3-point information  $I'(X; Y; Z|\{U_i\})$ .

**Out:** Partially oriented causal graph  $\mathcal{G}$  with edge orientation probabilities.

**3off2 Orientation / Propagation Step**

**sort** list of unshielded triples,  $\mathcal{L}_c = \{\langle X, Z, Y \rangle_{X \neq Y}\}$ , in decreasing order of their orientation/propagation probability initialized at 1/2 and computed from:

- (i.) Proposition 7, if  $I'(X; Y; Z|\{U_i\}) < 0$ , or
- (ii.) Proposition 8, if  $I'(X; Y; Z|\{U_i\}) > 0$

**repeat**

Take  $\langle X, Z, Y \rangle_{X \neq Y} \in \mathcal{L}_c$  with highest orientation / propagation probability  $> 1/2$ .

**if**  $I'(X; Y; Z|\{U_i\}) < 0$  **then**

**Orient**/propagate edge direction(s) to form a **v-structure**  $X \rightarrow Z \leftarrow Y$  with probabilities  $P_{X \rightarrow Z}$  and  $P_{Y \rightarrow Z}$  given by **Proposition 7**.

**else**

**Propagate** second edge direction to form a **non-v-structure**  $X \rightarrow Z \rightarrow Y$  assigning probability  $P_{Z \rightarrow Y}$  from **Proposition 8**.

**end**

Apply new orientation(s) and **sort** remaining list of unshielded triples  $\mathcal{L}_c \leftarrow \mathcal{L}_c \setminus \langle X, Z, Y \rangle_{X \neq Y}$  after **updating propagation probabilities**.

**until** no additional orient./propa. probability  $> 1/2$  ;

---

## 2.3 APPLICATIONS TO CAUSAL GRAPH BENCHMARKS

We have tested the 3off2 method on a range of benchmark networks of 50 nodes with up to 160 edges generated with the causal modeling tool Tetrad IV (<http://www.phil.cmu.edu/tetrad>). The average connectivity  $\langle k \rangle$  of these benchmark networks ranges between 1.6 to 6.4, and the average maximal in/out-degree between 3.2 to 8.8 (see Table S1 for a detailed description). The evaluation metrics are the Precision,  $Prec = TP/(TP + FP)$ , the Recall,  $Rec = TP/(TP + FN)$  and the  $F$ -score  $= 2Prec.Rec/(Prec + Rec)$ . However, in order to take into account the orientation/non-orientation of edges in the predicted networks and compare them with the CPDAG of the benchmark graphs, we define orientation-dependent counts as,  $TP' = TP - TP_{\text{misorient}}$  and  $FP' = FP + TP_{\text{misorient}}$ , where  $TP_{\text{misorient}}$  corresponds to all true positive edges of the skeleton with different orientation/non-orientation status as in the CPDAG reference.

The first methods used for comparison with 3off2 are the PC-stable algorithm (Colombo and Maathuis, 2014) with conservative (Ramsey, Spirtes, and Zhang, 2006) or majority orientation rules, implemented in the `pcalg` package (Kalisch et al., 2012; Kalisch and Bühlmann, 2008) and the hybrid method MMHC combining constraint-based skeleton and Bayesian orientation (Tsamardinos, Brown, and Aliferis, 2006), implemented in the `bnlearn` package (Scutari, 2010). Figs. 1-5 give the average CPDAG comparison results over 100 dataset replicates from 5 different benchmark networks (Table S1). The causal graphical models predicted by the 3off2 method are obtained using either the MDL/BIC or the NML complexities (see Supplementary Methods). Figs. S1-S6 provide additional results on the prediction of the network skeletons and execution times. The PC and MMHC results are shown, Figs. 1-5, for an independence test parameter  $\alpha = 0.1$ , as reducing  $\alpha$  tends to worsen the CPDAG F-score for benchmark networks with  $\langle k \rangle \geq 2.4$  (Figs. S7-S18). All in all, we found that 3off2 outperforms PC-stable on all tested datasets, Figs. 1-5, and to a lesser extent, MMHC especially on less sparse networks,  $\langle k \rangle \geq 2.4$ , Figs. 2-5.

Additional comparisons were obtained with Bayesian inference implemented in the `bnlearn` package (Scutari, 2010), using AIC, BDe and BIC/MDL scores and hill-climbing heuristics with 30 to 100 random restarts, Figs. S19-S30. 3off2 reaches equivalent or significantly better F-scores than Bayesian hill-climbing on relatively sparse benchmark networks,  $\langle k \rangle \leq 4.8$  (Figs. S19 & S23). In particular, 3off2 with MDL scores reaches one of the best F-scores on sparse networks (Figs. S19 & S20) and eventually better F-scores on large datasets for less sparse networks when combined to NML complexity (Figs. S21 & S22). For somewhat denser networks

( $\langle k \rangle \simeq 5$ ), the 3off2 F-score appears slightly lower than for Bayesian inference methods, Fig. S23, although it eventually becomes equivalent for large datasets ( $N \geq 1000$ ).

On denser networks ( $\langle k \rangle \geq 5 - 6$ ), Bayesian inference exhibits better F-scores than 3off2, in particular with AIC score, Fig. S24. However, the good performance with AIC strongly relies on its high Recall (but low Precision), due to its very small penalty term on large datasets, which makes it favor more complex networks (Figs. S24) but perform very poorly on sparse graphs (Figs. S19-S21). By contrast, the reconstruction of dense networks is impeded with the 3off2 scheme, as it is not always possible to uncover structural independencies,  $I(X; Y | \{U_{ij}\}_n) \simeq 0$ , in dense graphs through an ordered set  $\{U_{ij}\}_n$  with only positive conditional 3-point information,  $I'(X; Y; U_k | \{U_{ij}\}_{k-1}) > 0$ . Indeed in complex graphs, there are typically many indirect paths  $X \rightarrow U_j \rightarrow Y$  between unconnected node pairs  $(X, Y)$ . At the beginning of the pruning process, this is prone to suggest likely v-structures  $X \rightarrow Y \leftarrow U_j$ , instead of the correct non-v-structures,  $X \rightarrow U_j \rightarrow Y$  (for instance if  $I(X; U_j) \ll I(X; Y)$ ,  $I(X; U_j) \ll I(U_j; Y)$  and  $I(X; U_j) - I(X; U_j | Y) = I(X; Y; U_j) < 0$ , for all  $j$ ). Such elimination of *FN* edge  $X \rightarrow U_j$  and conservation of *FP*  $X \rightarrow Y$  tend to decrease both Precision and Recall, although 3off2 remains significantly more robust than PC and MMHC, Fig. 5. Besides, for most practical applications on real life data, interpretable causal models should remain relatively sparse and avoid to display multiple indirect paths between unconnected nodes.

Finally, 3off2 running times on these benchmark networks are similar to MMHC and Bayesian hill-climbing heuristic methods (with 100 restarts) and 10 to 100 times faster than PC for large datasets, Figs. S1-S30.

### 3 DISCUSSION

In this paper, we propose to combine constraint-based and score-based frameworks to improve network reconstruction. Earlier hybrid methods, including MMHC, have also attempted to exploit the best of these two types of inference approaches by combining the robustness of Bayesian scores with the attractive conceptual features of constraint-based approaches (Dash and Druzdzel, 1999; Tsamardinos, Brown, and Aliferis, 2006; Cano, Gomez-Olmedo, and Moral, 2008; Claassen and Heskes, 2012). In particular, (Dash and Druzdzel, 1999) have proposed to exploit an intrinsic weakness of the PC algorithm, its sensitivity to the order in which conditional independencies are tested on finite data, to rank these different order-dependent PC predictions with Bayesian scores. More recently, (Claassen and Heskes, 2012) have also combined constraint-based and Bayesian approaches to improve the reliability of causal inference. They proposed to use Bayesian scores to directly assess the reliability of conditional independen-

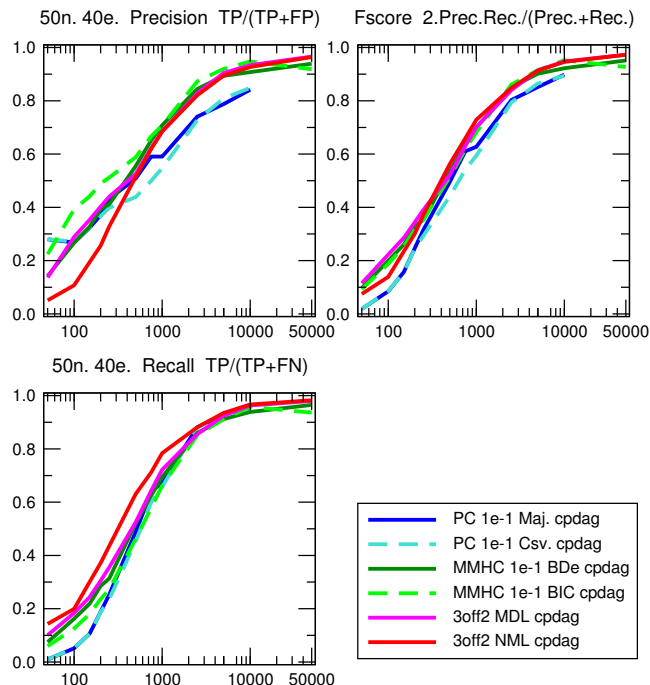


Figure 1: CPDAG comparison between 3off2, PC-stable and MMHC. 50 node, 40 edge benchmark networks generated using Tetrad.  $\langle k \rangle = 1.6$ ,  $\langle k_{\max}^{in} \rangle = 3.2$ ,  $\langle k_{\max}^{out} \rangle = 3.6$ . PC-stable benchmarks were tested up to  $N=10,000$  due to their sharp increase in execution time, see Figs. S7-S12.

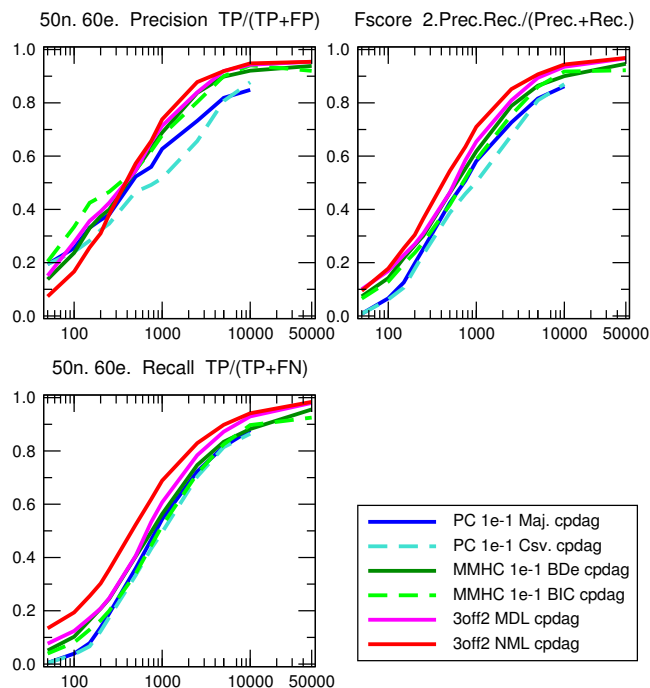


Figure 2: CPDAG comparison between 3off2, PC-stable and MMHC. 50 node, 60 edge benchmark networks generated using Tetrad.  $\langle k \rangle = 2.4$ ,  $\langle k_{\max}^{in} \rangle = 4.6$ ,  $\langle k_{\max}^{out} \rangle = 3.6$ . PC-stable benchmarks were tested up to  $N=10,000$  due to their sharp increase in execution time, see Figs. S7-S12.

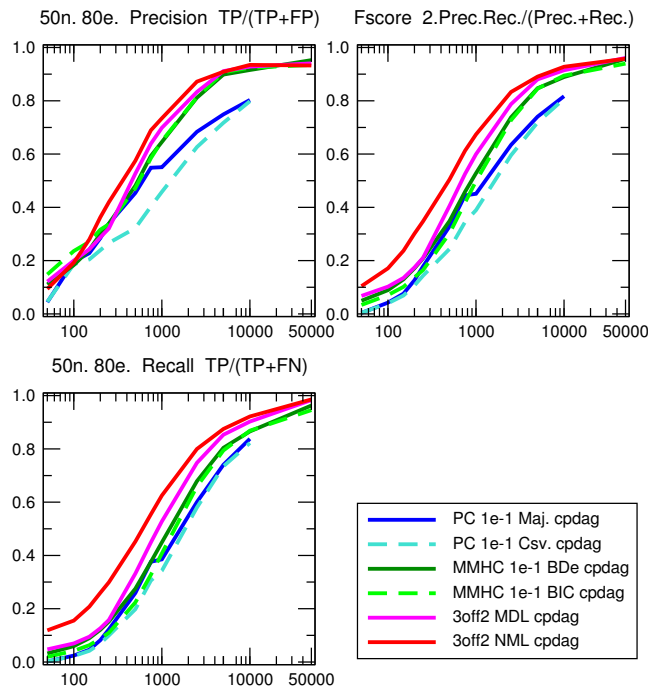


Figure 3: CPDAG comparison between 3off2, PC-stable and MMHC. 50 node, 80 edge benchmark networks generated using Tetrads.  $\langle k \rangle = 3.2$ ,  $\langle k_{\max}^{\text{in}} \rangle = 4.8$ ,  $\langle k_{\max}^{\text{out}} \rangle = 5.6$ . PC-stable benchmarks were tested up to  $N=10,000$  due to their sharp increase in execution time, see Figs. S7-S12.

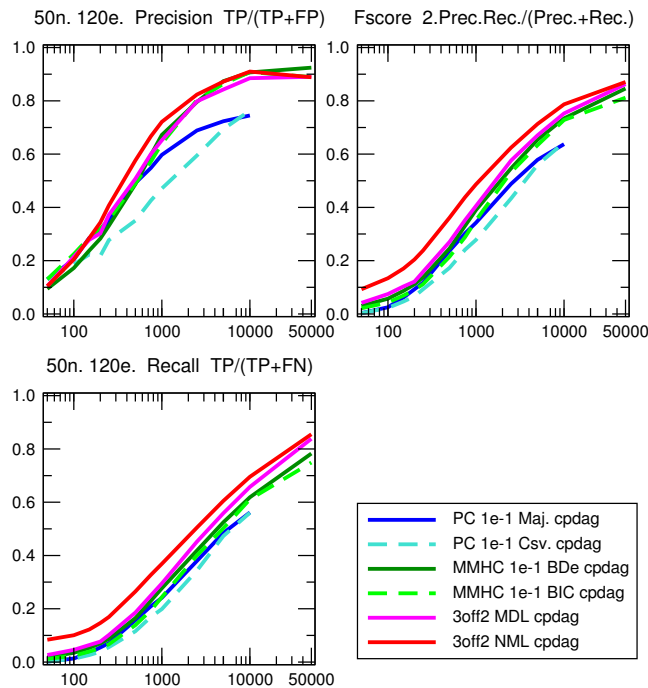


Figure 4: CPDAG comparison between 3off2, PC-stable and MMHC. 50 node, 120 edge benchmark networks generated using Tetrads.  $\langle k \rangle = 4.8$ ,  $\langle k_{\max}^{\text{in}} \rangle = 8.8$ ,  $\langle k_{\max}^{\text{out}} \rangle = 7.2$ . PC-stable benchmarks were tested up to  $N=10,000$  due to their sharp increase in execution time, see Figs. S7-S12.

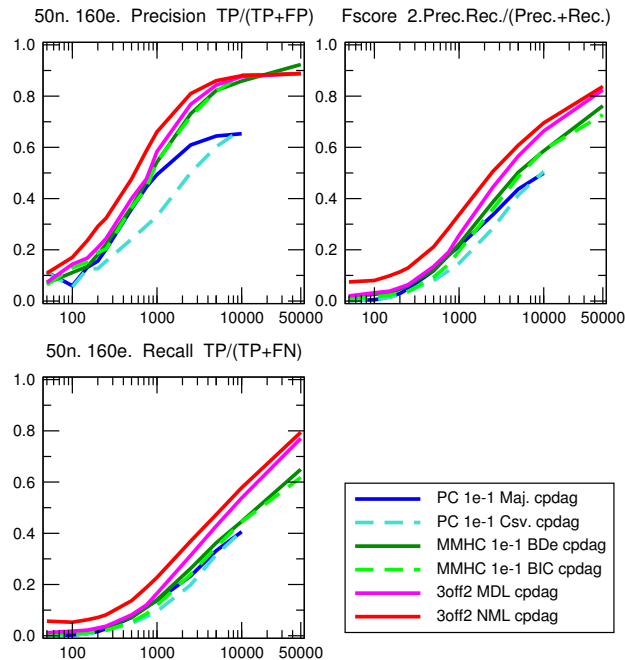


Figure 5: CPDAG comparison between 3off2, PC-stable and MMHC. 50 node, 160 edge benchmark networks generated using Tetrads.  $\langle k \rangle = 6.4$ ,  $\langle k_{\max}^{\text{in}} \rangle = 8.6$ ,  $\langle k_{\max}^{\text{out}} \rangle = 8.6$ . PC-stable benchmarks were tested up to  $N=10,000$  due to their sharp increase in execution time, see Figs. S7-S12.

cies by *summing* the likelihoods over compatible graphs. By contrast, we propose to use Bayesian scores to progressively uncover the best supported conditional independencies, by iteratively “taking off” the most likely indirect contributions of conditional 3-point information from every 2-point (mutual) information of the causal graph. In addition, using likelihood ratios (Eqs. 11 & 12) instead of likelihood sums (Claassen and Heskes, 2012) circumvents the need to score conditional independencies over a potentially intractable number of compatible graphs.

All in all, we found that 3off2 outperforms constraint-based, search-and-score and earlier hybrid methods on a range of benchmark networks, while displaying similar running times as hill-climbing heuristic methods.

### Acknowledgements

S.A. acknowledges support from Ministry of Higher Education and Research and Association pour la Recherche contre le Cancer (ARC). H.I. acknowledges funding from CNRS, Institut Curie and FPGG.

### References

Cano, A.; Gomez-Olmedo, M.; and Moral, S. 2008. A score based ranking of the edges for the pc algorithm. In *Proceedings of the European Workshop on Probabilistic Graphical Models (PGM)*, 41–48.

- Chickering, D. M. 2002. Learning equivalence classes of bayesian-network structures. *Journal of Machine Learning Research* 2:445–498.
- Claassen, T., and Heskes, T. 2012. A bayesian approach to constraint based causal inference. In *In Proc. of the 28th Conference on Uncertainty in Artificial Intelligence (UAI)*, 207–216. Morgan Kaufmann.
- Colombo, D., and Maathuis, M. H. 2014. Order-independent constraint-based causal structure learning. *Journal of Machine Learning Research* 15:3741–3782.
- Cooper, G. F., and Herskovits, E. 1992. A bayesian method for the induction of probabilistic networks from data. *Mach. Learn.* 9(4):309–347.
- Dash, D., and Druzdzel, M. J. 1999. A hybrid anytime algorithm for the construction of causal models from sparse data. In *Proceedings of the Fifteenth International Conference on Uncertainty in Artificial Intelligence*, 142–149. Morgan Kaufmann.
- de Campos, L. M. 2006. A scoring function for learning bayesian networks based on mutual information and conditional independence tests. *Journal of Machine Learning Research* 7:2149–2187.
- Hansen, M. H., and Yu, B. 2001. Model selection and the principle of minimum description length. *Journal of the American Statistical Association* 96:746–774.
- Heckerman, D.; Geiger, D.; and Chickering, D. M. 1995. Learning Bayesian Networks: The Combination of Knowledge and Statistical Data. *Machine Learning* 20(3):197–243. Available as Technical Report MSR-TR-94-09.
- Kalisch, M., and Bühlmann, P. 2008. Robustification of the pc-algorithm for directed acyclic graphs. *Journal Of Computational And Graphical Statistics* 17(4):773–789.
- Kalisch, M.; Mächler, M.; Colombo, D.; Maathuis, M. H.; and Bühlmann, P. 2012. Causal inference using graphical models with the r package pcalg. *Journal of Statistical Software* 47(11):1–26.
- Koller, D., and Friedman, N. 2009. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press.
- Kontkanen, P., and Myllymäki, P. 2007. A linear-time algorithm for computing the multinomial stochastic complexity. *Inf. Process. Lett.* 103(6):227–233.
- Meek, C. 1995. Causal inference and causal explanation with background knowledge. In *Proceedings of Eleventh Conference on Uncertainty in Artificial Intelligence*, Montreal, QU. Morgan Kaufmann. 403–418.
- Pearl, J., and Verma, T. 1991. A theory of inferred causation. In *In Knowledge Representation and Reasoning: Proc. of the Second Int. Conf.* 441–452.
- Pearl, J. 2009. *Causality: models, reasoning and inference*. Cambridge University Press, 2nd edition.
- Ramsey, J.; Spirtes, P.; and Zhang, J. 2006. Adjacency-faithfulness and conservative causal inference. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*, UAI, 401–408. Oregon, USA: AUAI Press.
- Rebane, G., and Pearl, J. 1988. The recovery of causal poly-trees from statistical data. *Int. J. Approx. Reasoning* 2(3):341.
- Reichenbach, H. 1956. *The Direction of Time*. California library reprint series. University of California Press.
- Rissanen, J., and Tabus, I. 2005. Kolmogorovs structure function in mdl theory and lossy data compression. In *Adv. Min. Descrip. Length Theory Appl.* MIT Press. Chap. 10.
- Rissanen, J. 1978. Modeling by shortest data description. *Automatica* vol. 14:465–471.
- Roos, T.; Silander, T.; Kontkanen, P.; and Myllymäki, P. 2008. Bayesian network structure learning using factorized nml universal models. In *Proc. 2008 Information Theory and Applications Workshop (ITA-2008)*. IEEE Press. invited paper.
- Sanov, I. 1957. On the probability of large deviations of random variables. *Mat. Sbornik* 42:11–44.
- Scutari, M. 2010. Learning Bayesian Networks with the bnlearn R Package. *Journal of Statistical Software* 35(3):1–22.
- Shtarkov, Y. M. 1987. Universal sequential coding of single messages. *Problems of Information Transmission (Translated from)* 23(3):3–17.
- Spirtes, P., and Glymour, C. 1991. An algorithm for fast recovery of sparse causal graphs. *Social Science Computer Review* 9:62–72.
- Spirtes, P.; Glymour, C.; and Scheines, R. 2000. *Causation, Prediction, and Search*. The MIT Press, Cambridge, Massachusetts, 2nd edition.
- Tsamardinos, I.; Brown, L. E.; and Aliferis, C. F. 2006. The Max-Min Hill-Climbing Bayesian Network Structure Learning Algorithm. *Machine Learning* 65(1):31–78.

---

# Are You Doing What I Think You Are Doing? Criticising Uncertain Agent Models

---

**Stefano V. Albrecht**  
School of Informatics  
University of Edinburgh  
Edinburgh EH8 9AB, UK  
s.v.albrecht@sms.ed.ac.uk

**Subramanian Ramamoorthy**  
School of Informatics  
University of Edinburgh  
Edinburgh EH8 9AB, UK  
s.ramamoorthy@ed.ac.uk

## Abstract

The key for effective interaction in many multi-agent applications is to reason explicitly about the behaviour of other agents, in the form of a *hypothesised* behaviour. While there exist several methods for the construction of a behavioural hypothesis, there is currently no universal theory which would allow an agent to contemplate the correctness of a hypothesis. In this work, we present a novel algorithm which decides this question in the form of a frequentist hypothesis test. The algorithm allows for multiple metrics in the construction of the test statistic and learns its distribution during the interaction process, with asymptotic correctness guarantees. We present results from a comprehensive set of experiments, demonstrating that the algorithm achieves high accuracy and scalability at low computational costs.

## 1 INTRODUCTION

A common difficulty in many multiagent systems is the fact that the behaviour of other agents may be initially unknown. Important examples include adaptive user interfaces, robotic elderly assistance, and electronic markets. Often, the key for effective interaction in such systems is to reason explicitly about the behaviour of other agents, typically in the form of a *hypothesised* behaviour which makes predictions about future actions based on a given interaction history.

A number of methods have been studied for the construction of behavioural hypotheses. One method is to use opponent modelling techniques to learn a behaviour from the interaction history. Two well-known examples include fictitious play (Brown, 1951) and case-based reasoning (Gilboa and Schmeidler, 2001), as well as their many variants. Another method is to maintain a set of possible action policies, called types, over which a posterior belief is computed based on the interaction history (Albrecht and Ramamoorthy, 2014; Gmytrasiewicz and Doshi, 2005). The hypothesis is then

obtained by using the posterior to mix the types. Related methods have been studied in the plan recognition literature (Carberry, 2001; Charniak and Goldman, 1993).

The learned behaviours (or models) of these methods can be viewed as hypotheses because they are eventually either true or false (subject to the various assumptions they are based on), and because they are *testable*. Thus, the following is a natural question: given an interaction history  $H$  and a hypothesis  $\pi^*$  for the behaviour of an agent, does the agent indeed behave according to  $\pi^*$ ? There are several ways in which an answer to this question could be utilised. For instance, if we persistently reject the hypothesis  $\pi^*$ , we may construct an alternative hypothesis or resort to some default plan of action (such as a “maximin” strategy).

Unfortunately, the above methods for hypothesis construction do not provide an answer to this question. Some opponent modelling methods use goodness-of-fit measures (e.g. those that rely on maximum likelihood estimation), but these measures describe how well the model fits the data (i.e. interaction history) and not necessarily if the model is correct. Similarly, the posterior belief in the type-based approach quantifies the relative likelihood of types (relative to a set of alternative types) but not the *correctness* of types.

To illustrate the source of difficulty, consider the below excerpt of an interaction process between two agents which can choose from three actions. The columns show, respectively, the current time  $t$  of the interaction, the actions chosen by the agents at time  $t$ , and agent 1’s hypothesised probabilities with which agent 2 would choose its actions at time  $t$ , based on the prior interaction history.

| $t$ | $(a_1^t, a_2^t)$ | $\pi_2^*$    |
|-----|------------------|--------------|
| 1   | (1, 2)           | (.3, .1, .6) |
| 2   | (3, 1)           | (.2, .3, .5) |
| 3   | (2, 3)           | (.7, .1, .2) |
| 4   | (2, 3)           | (.0, .4, .6) |
| 5   | (1, 2)           | (.4, .2, .4) |

Assuming that the process continues in this fashion, and without any restrictions on the behaviour of agent 2, how

should agent 1 decide whether or not to reject its hypothesis about the behaviour of agent 2?

A natural way to address this question is to compute some kind of *score* from the information given in the above table, and to compare this score with some manually chosen rejecting threshold. A prominent example of such a score is the empirical frequency distribution (Conitzer and Sandholm, 2007; Foster and Young, 2003). While the simplicity of this method is appealing, there are two significant problems: (1) it is far from trivial to devise a scoring scheme that reliably quantifies “correctness” of hypotheses (for instance, an empirical frequency distribution taken over all past actions would be insufficient in the above example since the hypothesised action distributions are changing), and (2) it is unclear how one should choose the threshold parameter for any given scoring scheme.

In this work, we present an efficient algorithm which decides this question in the form of a frequentist hypothesis test. The algorithm addresses (1) by allowing for multiple scoring criteria in the construction of the test statistic, with the intent of obtaining an overall more reliable scoring scheme. The distribution of the test statistic is then learned during the interaction process, and we show that the learning is asymptotically correct. Finally, analogous to standard frequentist testing, the hypothesis is rejected at a given point in time if the resulting  $p$ -value is below some “significance level”. This eliminates (2) by providing a uniform semantic for rejection that is invariant to the employed scoring scheme. We present a comprehensive set of experiments, demonstrating that our algorithm achieves high accuracy and scalability at low computational costs.

Of course, there is a long-standing debate on the role of statistical hypothesis tests and quantities such as  $p$ -values (e.g. Gelman and Shalizi, 2013; Berger and Sellke, 1987; Cox, 1977). The usual consensus is that  $p$ -values should be combined with other forms of evidence to reach a final conclusion (Fisher, 1935), and this is the view we adopt as well. In this sense, our method may be used as part of a larger machinery to decide the truth of a hypothesis.

## 2 RELATED WORK

In addition to the related works mentioned in the previous section, there are a number of other related research areas:

There exists a large body of literature on what is often referred to as *model criticism* (e.g. Bayarri and Berger, 2000; Meng, 1994; Rubin, 1984; Box, 1980). Model criticism attempts to answer the following question: given a data set  $D$  and model  $M$ , could  $D$  have been generated by  $M$ ? This is analogous to our question, in which  $D$  is a sequence of observed actions of some agent and  $M$  is a hypothesised behaviour for that agent. However, in contrast to our work, model criticism usually assumes that the data are indepen-

dent and identically distributed, which is not the case in the interactive settings we consider.

A related problem, sometimes referred to as *identity testing*, is to test if a given sequence of data was generated by some given stochastic process (Ryabko and Ryabko, 2008; Basawa and Scott, 1977). Instead of independent and identical distributions, this line of work assumes other properties such as stationarity and ergodicity. Unfortunately, these assumptions are also unlikely in interaction processes, and the proposed solutions are very costly.

Model criticism and identity testing are not to be confused with *model selection*, in which two or more alternative models are under consideration (e.g. Vehtari and Ojanen, 2012). Similarly, we do not consider alternative hypotheses. However, our method can be applied individually to multiple hypotheses, or the hypotheses may be fused into a single hypothesis using a posterior belief (Albrecht and Ramamoorthy, 2014; Gmytrasiewicz and Doshi, 2005).

Another related problem is that of *model checking*, which attempts to verify that a given system (or model) satisfies certain formal properties (Clarke et al., 1999). Recently, Albrecht and Ramamoorthy (2014) applied the concept of probabilistic bisimulation (Larsen and Skou, 1991) to the question of “incorrect” hypotheses and showed that a certain form of optimality is preserved if a bisimulation relation exists. However, their work is not concerned with establishing whether or not a given behavioural hypothesis is correct, and their analysis is performed *before* any interaction.

Our method can be viewed as *passive* in the sense that it does not actively probe different aspects of the hypothesis, and we show in Section 5 that this can be a drawback. This is in contrast to methods such as (Carmel and Markovitch, 1999), which promote active exploration. However, this exploration comes at high computational costs and limits the structure of hypotheses, such as deterministic finite state machines. On the other hand, our method has low computational costs and leaves the structure of the hypothesis open.

## 3 PRELIMINARIES

We consider a sequential interaction process with  $m$  agents. The process begins at time  $t = 0$ . At each time  $t$ , each agent  $i \in \{1, \dots, m\}$  receives a signal  $s_i^t$  and chooses an action  $a_i^t$  from a finite set of actions  $A_i$ . (Agents choose actions simultaneously.) The process continues in this fashion indefinitely or until some termination criterion is satisfied.

The signal  $s_i^t$  specifies information that agent  $i$  receives at time  $t$  and may in general be the result of a random variable over past actions and signals. For example,  $s_i^t$  may be a discrete system state and its dynamics may be described by some stochastic transition function. Note that we allow for asymmetric information (i.e.  $s_i^t \neq s_j^t$ ). For example,  $s_i^t$  may include a private payoff for agent  $i$ . In this work, we leave

the precise structure and dynamics of  $s_i^t$  open.

We assume that each agent  $i$  can choose actions  $a_i^t$  based on the entire interaction history  $H_i^t = (s_i^0, a^0, s_i^1, a^1, \dots, s_i^t)$ , where  $a^\tau = (a_1^\tau, \dots, a_m^\tau)$  is the tuple of actions taken by the agents at time  $\tau$ . Formally, each agent  $i$  has a *behaviour*  $\pi_i$  which assigns a probability distribution over actions  $A_i$  given a history  $H_i^t$ , denoted  $\pi_i(H_i^t)$ . We use  $\Pi_i$  to denote the infinite and uncountable space of all such behaviours. Note that a behaviour may implement any kind of logic, and it is useful to think of it as a black-box programme.

Given two agents  $i$  and  $j$ , we use  $\Pi_j^i$  to denote  $i$ 's *hypothesis space* for  $j$ 's behaviours. The difference between  $\Pi_j^i$  and  $\Pi_j$  is that  $\pi_j^* \in \Pi_j^i$  are defined over  $H_i^t$  while  $\pi_j \in \Pi_j$  are defined over  $H_j^t$ . Since we allow for asymmetric information, any information that is contained in  $s_j^t$  but not in  $s_i^t$ , denoted  $s_{j-i}^t$ , becomes part of the hypothesis space  $\Pi_j^i$ . For example, if  $s_{j-i}^t$  contains a private payoff for  $j$ ,  $i$  can hypothesise a payoff as part of its hypothesis for  $j$ 's behaviour.

Defining a behavioural hypothesis  $\pi_j^* \in \Pi_j^i$  as a function  $\pi_j^*(H_i^t)$  has two implicit assumptions: firstly, it assumes knowledge of  $A_j$ , and secondly, it assumes that the information in  $s_{j-i}^t$  is a (deterministic) function of  $H_i^t$ . If, on the other hand, we allowed  $s_{j-i}^t$  to be stochastic (i.e. a random variable over the interaction history), we would in addition have to hypothesise the random outcome of  $s_{j-i}^t$ . In other words,  $\pi_j^*(H_i^t)$  would itself be a random variable, which is outside the scope of this work.

## 4 A METHOD FOR BEHAVIOURAL HYPOTHESIS TESTING

Let  $i$  denote our agent and let  $j$  denote another agent. Moreover, let  $\pi_j^* \in \Pi_j^i$  denote our hypothesis for  $j$ 's behaviour and let  $\pi_j \in \Pi_j$  denote  $j$ 's true behaviour. The central question we ask is if  $\pi_j^* = \pi_j$ ?

Unfortunately, since we do not know  $\pi_j$ , we cannot directly answer this question. However, at each time  $t$ , we know  $j$ 's past actions  $\mathbf{a}_j^t = (a_j^0, \dots, a_j^{t-1})$  which were generated by  $\pi_j$ . If we use  $\pi_j^*$  to generate a vector  $\hat{\mathbf{a}}_j^t = (\hat{a}_j^0, \dots, \hat{a}_j^{t-1})$ , where  $\hat{a}_j^\tau$  is sampled using  $\pi_j^*(H_i^\tau)$ , we can formulate the related two-sample problem of whether  $\mathbf{a}_j^t$  and  $\hat{\mathbf{a}}_j^t$  were generated from the same behaviour, namely  $\pi_j^*$ .

In this section, we propose a general and efficient algorithm to decide this problem. At its core, the algorithm computes a frequentist  $p$ -value

$$p = P\left(|T(\hat{\mathbf{a}}_j^t, \hat{\mathbf{a}}_j^t)| \geq |T(\mathbf{a}_j^t, \hat{\mathbf{a}}_j^t)|\right) \quad (1)$$

where  $\hat{\mathbf{a}}_j^t \sim \delta^t(\pi_j^*) = (\pi_j^*(H_i^0), \dots, \pi_j^*(H_i^{t-1}))$ . The value of  $p$  corresponds to the probability with which we expect to observe a test statistic at least as extreme as  $T(\mathbf{a}_j^t, \hat{\mathbf{a}}_j^t)$ , under the null-hypothesis  $\pi_j^* = \pi_j$ . Thus, we reject  $\pi_j^*$  if  $p$  is below some ‘‘significance level’’  $\alpha$ .

---

### Algorithm 1

---

- 1: **Input:** history  $H_i^t$  (including observed action  $a_j^{t-1}$ )
  - 2: **Output:**  $p$ -value (reject  $\pi_j^*$  if  $p$  below some threshold  $\alpha$ )
  - 3: **Parameters:** hypothesis  $\pi_j^*$ ; score functions  $z_1, \dots, z_K$ ;  $N > 0$
  - 4: // Expand action vectors
  - 5: Set  $\mathbf{a}_j^t \leftarrow \langle \mathbf{a}_j^{t-1}, a_j^{t-1} \rangle$
  - 6: Sample  $\hat{a}_j^{t-1} \sim \pi_j^*(H_i^{t-1})$ ; set  $\hat{\mathbf{a}}_j^t \leftarrow \langle \hat{\mathbf{a}}_j^{t-1}, \hat{a}_j^{t-1} \rangle$
  - 7: **for**  $n = 1, \dots, N$  **do**
  - 8:   Sample  $\tilde{a}_j^{t-1} \sim \pi_j^*(H_i^{t-1})$ ; set  $\tilde{\mathbf{a}}_j^{t,n} \leftarrow \langle \tilde{\mathbf{a}}_j^{t-1,n}, \tilde{a}_j^{t-1} \rangle$
  - 9: // Fit skew-normal distribution  $f$
  - 10: **if** update parameters? **then**
  - 11:   Compute  $D \leftarrow \{T(\tilde{\mathbf{a}}_j^{t,n}, \hat{\mathbf{a}}_j^t) \mid n = 1, \dots, N\}$
  - 12:   Fit  $\xi, \omega, \beta$  to  $D$ , e.g. using (12)
  - 13:   Find mode  $\mu$  from  $\xi, \omega, \beta$
  - 14: // Compute  $p$ -value
  - 15:   Compute  $q \leftarrow T(\mathbf{a}_j^t, \hat{\mathbf{a}}_j^t)$  using (2)/(5)
  - 16: **return**  $p \leftarrow f(q \mid \xi, \omega, \beta) / f(\mu \mid \xi, \omega, \beta)$
- 

In the following subsections, we describe the test statistic  $T$  and its asymptotic properties, and how our algorithm learns the distribution of  $T(\tilde{\mathbf{a}}_j^t, \hat{\mathbf{a}}_j^t)$ . A summary of the algorithm is given in Algorithm 1.

#### 4.1 TEST STATISTIC

We follow the general approach outlined in Section 1 by which we compute a *score* from a vector of actions and their hypothesised distributions. Formally, we define a *score function* as  $z : (A_j)^t \times \Delta(A_j)^t \rightarrow \mathbb{R}$ , where  $\Delta(A_j)$  is the set of all probability distributions over  $A_j$ . Thus,  $z(\mathbf{a}_j^t, \delta^t(\pi_j^*))$  is the score for observed actions  $\mathbf{a}_j^t$  and hypothesised distributions  $\delta^t(\pi_j^*)$ , and we sometimes abbreviate this to  $z(\mathbf{a}_j^t, \pi_j^*)$ . We use  $Z$  to denote the space of all score functions.

Given a score function  $z$ , we define the test statistic  $T$  as

$$T(\tilde{\mathbf{a}}_j^t, \hat{\mathbf{a}}_j^t) = \frac{1}{t} \sum_{\tau=1}^t T_\tau(\tilde{\mathbf{a}}_j^\tau, \hat{\mathbf{a}}_j^\tau) \quad (2)$$

$$T_\tau(\tilde{\mathbf{a}}_j^\tau, \hat{\mathbf{a}}_j^\tau) = z(\tilde{\mathbf{a}}_j^\tau, \pi_j^*) - z(\hat{\mathbf{a}}_j^\tau, \pi_j^*) \quad (3)$$

where  $\tilde{\mathbf{a}}_j^\tau$  and  $\hat{\mathbf{a}}_j^\tau$  are the  $\tau$ -prefixes of  $\tilde{\mathbf{a}}_j^t$  and  $\hat{\mathbf{a}}_j^t$ , respectively.

In this work, we assume that  $z$  is provided by the user. While formally unnecessary (in the sense that our analysis does not require it), we find it a useful design guideline to interpret a score as a kind of likelihood, such that higher scores suggest higher likelihood of  $\pi_j^*$  being correct. Under this interpretation, a minimum requirement for  $z$  should be that it is *consistent*, such that, for any  $t > 0$  and  $\pi_j^* \in \Pi_j^i$ ,

$$\pi_j^* \in \Pi^z = \arg \max_{\pi_j^* \in \Pi_j^i} \mathbb{E}_{\mathbf{a}_j^t \sim \delta^t(\pi_j^*)} [z(\mathbf{a}_j^t, \pi_j^*)] \quad (4)$$

where  $\mathbb{E}_\eta$  denotes the expectation under  $\eta$ . This ensures



that if the null-hypothesis  $\pi_j^* = \pi_j$  is true, then the score  $z(\mathbf{a}_j^t, \pi_j^*)$  is maximised on expectation.

Ideally, we would like a score function  $z$  which is *perfect* in that it is consistent and  $|\Pi^z| = 1$ . This means that  $\pi_j^*$  can maximise  $z(\mathbf{a}_j^t, \pi_j^*)$  (where  $\mathbf{a}_j^t \sim \delta^t(\pi_j)$ ) *only* if  $\pi_j^* = \pi_j$ . Unfortunately, it is unclear if such a score function exists for the general case and how it should look. Even if we restrict the behaviours agents may exhibit, it can still be difficult to find a perfect score function. On the other hand, it is a relatively simple task to specify a small set of score functions  $z_1, \dots, z_K$  which are consistent but imperfect. (Examples are given in Section 5.) Given that these score functions are consistent, we know that the cardinality  $|\cap_k \Pi^{z_k}|$  can only monotonically decrease. Therefore, it seems a reasonable approach to combine multiple imperfect score functions in an attempt to approximate a perfect score function.

Of course, we could simply define  $z$  as a linear (or otherwise) combination of  $z_1, \dots, z_K$ . However, this approach is at risk of losing information from the individual scores, e.g. due to commutativity and other properties of the combination. Thus, we instead propose to compare the scores individually. Given score functions  $z_1, \dots, z_K \in Z$  which are all bounded by the same interval  $[a, b] \subset \mathbb{R}$ , we redefine  $T_\tau$  to

$$T_\tau(\tilde{\mathbf{a}}_j^\tau, \hat{\mathbf{a}}_j^\tau) = \sum_{k=1}^K w_k (z_k(\tilde{\mathbf{a}}_j^\tau, \pi_j^*) - z_k(\hat{\mathbf{a}}_j^\tau, \pi_j^*)) \quad (5)$$

where  $w_k \in \mathbb{R}$  is a weight for score function  $z_k$ . In this work, we set  $w_k = \frac{1}{K}$ . (We also experiment with alternative weighting schemes in Section 5.) However, we believe that  $w_k$  may serve as an interface for useful modifications of our algorithm. For example, Yue et al. (2010) compute weights to increase the power of their specific hypothesis tests.

## 4.2 ASYMPTOTIC PROPERTIES

The vectors  $\mathbf{a}_j^t$  and  $\hat{\mathbf{a}}_j^t$  are constructed iteratively. That is, at time  $t$ , we observe agent  $j$ 's past action  $a_j^{t-1}$ , which was generated from  $\pi_j(H_j^{t-1})$ , and set  $\mathbf{a}_j^t = \langle \mathbf{a}_j^{t-1}, a_j^{t-1} \rangle$ . At the same time, we sample an action  $\hat{a}_j^{t-1}$  using  $\pi_j^*(H_i^{t-1})$  and set  $\hat{\mathbf{a}}_j^t = \langle \hat{\mathbf{a}}_j^{t-1}, \hat{a}_j^{t-1} \rangle$ . Assuming the null-hypothesis  $\pi_j^* = \pi_j$ , will  $T(\mathbf{a}_j^t, \hat{\mathbf{a}}_j^t)$  converge in the process?

Unfortunately,  $T$  might not converge. This may seem surprising at first glance given that  $a_j^{t-1}, \hat{a}_j^{t-1}$  have the same distribution  $\pi_j(H_j^{t-1}) = \pi_j^*(H_i^{t-1})$ , since  $\mathbb{E}_{x,y \sim \psi} [x - y] = 0$  for any distribution  $\psi$ . However, there is a subtle but important difference: while  $a_j^{t-1}, \hat{a}_j^{t-1}$  have the same distribution,  $z_k(\mathbf{a}_j^t, \pi_j^*)$  and  $z_k(\hat{\mathbf{a}}_j^t, \pi_j^*)$  may have arbitrarily different distributions. This is because these scores may depend on the entire prefix vectors  $\mathbf{a}_j^{t-1}$  and  $\hat{\mathbf{a}}_j^{t-1}$ , respectively, which means that their distributions may be different if  $\mathbf{a}_j^{t-1} \neq \hat{\mathbf{a}}_j^{t-1}$ . Fortunately, our algorithm does not require  $T$  to converge because it learns the distribution of  $T$  during the interaction process, as we will discuss in Section 4.3.

Interestingly, while  $T$  may not converge, it can be shown that the fluctuation of  $T$  is eventually normally distributed, for any set of score functions  $z_1, \dots, z_K$  with bound  $[a, b]$ . Formally, let  $\mathbb{E}[T_\tau(\mathbf{a}_j^\tau, \hat{\mathbf{a}}_j^\tau)]$  and  $\text{Var}[T_\tau(\mathbf{a}_j^\tau, \hat{\mathbf{a}}_j^\tau)]$  denote the finite expectation and variance of  $T_\tau(\mathbf{a}_j^\tau, \hat{\mathbf{a}}_j^\tau)$ , where it is irrelevant if  $\mathbf{a}_j^\tau, \hat{\mathbf{a}}_j^\tau$  are sampled directly from  $\delta^\tau(\pi_j^*)$  or generated iteratively as prescribed above. Furthermore, let  $\sigma_t^2 = \sum_{\tau=1}^t \text{Var}[T_\tau(\mathbf{a}_j^\tau, \hat{\mathbf{a}}_j^\tau)]$  denote the cumulative variance. Then, the standardised stochastic sum

$$\frac{1}{\sigma_t} \sum_{\tau=1}^t T_\tau(\mathbf{a}_j^\tau, \hat{\mathbf{a}}_j^\tau) - \mathbb{E}[T_\tau(\mathbf{a}_j^\tau, \hat{\mathbf{a}}_j^\tau)] \quad (6)$$

will converge in distribution to the standard normal distribution as  $t \rightarrow \infty$ . Thus,  $T$  is normally distributed as well.

To see this, first recall that the standard central limit theorem requires the random variables  $T_\tau$  to be independent and identically distributed. In our case,  $T_\tau$  are independent in that the random outcome of  $T_\tau$  has no effect on the outcome of  $T_{\tau'}$ . However,  $T_\tau$  and  $T_{\tau'}$  depend on different action sequences, and may therefore have different distributions. Hence, we have to show an additional property, commonly known as *Lyapunov's condition* (e.g. Fischer, 2010), which states that there exists a positive integer  $d$  such that

$$\lim_{t \rightarrow \infty} \frac{\hat{\sigma}_t^{2+d}}{\sigma_t^{2+d}} = 0, \text{ with} \quad (7)$$

$$\hat{\sigma}_t^{2+d} = \sum_{\tau=1}^t \mathbb{E} \left[ |T_\tau(\mathbf{a}_j^\tau, \hat{\mathbf{a}}_j^\tau) - \mathbb{E}[T_\tau(\mathbf{a}_j^\tau, \hat{\mathbf{a}}_j^\tau)]|^{2+d} \right]. \quad (8)$$

Since  $z_k$  are bounded, we know that  $T_\tau$  are bounded. Hence, the summands in (8) are uniformly bounded, say by  $U$  for brevity. Setting  $d = 1$ , we obtain

$$\lim_{t \rightarrow \infty} \frac{\hat{\sigma}_t^3}{\sigma_t^3} \leq \frac{U \hat{\sigma}_t^2}{\sigma_t^3} = \frac{U}{\sigma_t} \quad (9)$$

The last part goes to zero if  $\sigma_t \rightarrow \infty$ , and hence Lyapunov's condition holds. If, on the other hand,  $\sigma_t$  converges, then this means that the variance of  $T_\tau$  is zero from some point onward (or that it has an appropriate convergence to zero). In this case,  $\pi_j^*$  will prescribe deterministic action choices for agent  $j$ , and a statistical analysis is no longer necessary.

## 4.3 LEARNING THE TEST DISTRIBUTION

Given that  $T$  is eventually normal, it may seem reasonable to compute (1) using a normal distribution whose parameters are fitted during the interaction. However, this fails to recognise that the distribution of  $T$  is shaped *gradually* over an extended time period, and that the fluctuation around  $T$  can be heavily skewed in either direction until convergence to a normal distribution emerges. Thus, a normal distribution may be a poor fit during this shaping period.

What is needed is a distribution which can represent any normal distribution, and which is flexible enough to faithfully

represent the gradual shaping. One distribution which has these properties is the *skew-normal distribution* (Azzalini, 1985; O’Hagan and Leonard, 1976). Given the PDF  $\phi$  and CDF  $\Phi$  of the standard normal distribution, the skew-normal PDF is defined as

$$f(x | \xi, \omega, \beta) = \frac{2}{\omega} \phi\left(\frac{x - \xi}{\omega}\right) \Phi\left(\beta \left(\frac{x - \xi}{\omega}\right)\right) \quad (10)$$

where  $\xi \in \mathbb{R}$  is the location parameter,  $\omega \in \mathbb{R}^+$  is the scale parameter, and  $\beta \in \mathbb{R}$  is the shape parameter. Note that this reduces to the normal PDF for  $\beta = 0$ , in which case  $\xi$  and  $\omega$  correspond to the mean and standard deviation, respectively. Hence, the normal distribution is a sub-class of the skew-normal distribution.

Our algorithm learns the shifting parameters of  $f$  during the interaction process, using a simple but effective sampling procedure. Essentially, we use  $\pi_j^*$  to iteratively generate  $N$  additional action vectors  $\hat{\mathbf{a}}_j^{t,1}, \dots, \hat{\mathbf{a}}_j^{t,N}$  in the exact same way as  $\hat{\mathbf{a}}_j^t$ . The vectors  $\hat{\mathbf{a}}_j^{t,n}$  are then mapped into data points

$$D = \left\{ T(\hat{\mathbf{a}}_j^{t,n}, \hat{\mathbf{a}}_j^t) \mid n = 1, \dots, N \right\} \quad (11)$$

which are used to estimate the parameters  $\xi, \omega, \beta$  by minimising the negative log-likelihood

$$N \log(\omega) - \sum_{x \in D} \log \phi\left(\frac{x - \xi}{\omega}\right) + \log \Phi\left(\beta \left(\frac{x - \xi}{\omega}\right)\right) \quad (12)$$

whilst ensuring that  $\omega$  is positive. An alternative is the method-of-moments estimator, which can also be used to obtain initial values for (12). Note that it is usually unnecessary to estimate the parameters at every point in time. Rather, it seems reasonable to update the parameters less frequently as the amount of evidence (i.e. observed actions) grows.

Given the asymmetry of the skew-normal distribution, the semantics of “as extreme as” in (1) may no longer be obvious (e.g. is this with respect to the mean or mode?). In addition, the usual tail-area calculation of the  $p$ -value requires the CDF, but there is no closed form for the skew-normal CDF and approximating it is rather cumbersome. To circumvent these issues, we approximate the  $p$ -value as

$$p \approx \frac{f(T(\hat{\mathbf{a}}_j^t, \hat{\mathbf{a}}_j^t) | \xi, \omega, \beta)}{f(\mu | \xi, \omega, \beta)} \quad (13)$$

where  $\mu$  is the mode of the fitted skew-normal distribution. This avoids the asymmetry issue and is easier to compute.

## 5 EXPERIMENTS

We conducted a comprehensive set of experiments to investigate the accuracy (correct and incorrect rejection), scalability (with number of actions), and sampling complexity of

our algorithm. The following three score functions and their combinations were used:

$$z_1(\mathbf{a}_j^t, \pi_j^*) = \frac{1}{t} \sum_{\tau=0}^{t-1} \frac{\pi_j^*(H_i^\tau)[a_j^\tau]}{\max_{a_j \in A_j} \pi_j^*(H_i^\tau)[a_j]}$$

$$z_2(\mathbf{a}_j^t, \pi_j^*) = \frac{1}{t} \sum_{\tau=0}^{t-1} 1 - \mathbb{E}_{\pi_j^*(H_i^\tau)}^{a_j \sim} |\pi_j^*(H_i^\tau)[a_j^\tau] - \pi_j^*(H_i^\tau)[a_j]|$$

$$z_3(\mathbf{a}_j^t, \pi_j^*) = \sum_{a_j \in A_j} \min \left[ \frac{1}{t} \sum_{\tau=0}^{t-1} [a_j^\tau = a_j]_1, \frac{1}{t} \sum_{\tau=0}^{t-1} \pi_j^*(H_i^\tau)[a_j] \right]$$

where  $[b]_1 = 1$  if  $b$  is true and 0 otherwise. Note that  $z_1, z_3$  are generally consistent (cf. Section 4.1), while  $z_2$  is consistent for  $|A_j| = 2$  but not necessarily for  $|A_j| > 2$ . Furthermore,  $z_1, z_2, z_3$  are all imperfect. The score function  $z_3$  is based on the empirical frequency distribution (cf. Section 1).

The parameters of the test distribution (cf. Section 4.3) were estimated less frequently as  $t$  increased. The first estimation was performed at time  $t = 1$  (i.e. after observing one action). After estimating the parameters at time  $t$ , we waited  $\lfloor \sqrt{t} \rfloor - 1$  time steps until the parameters were re-fitted. Throughout our experiments, we used a significance level of  $\alpha = 0.01$  (i.e. reject  $\pi_j^*$  if the  $p$ -value is below 0.01).

### 5.1 RANDOM BEHAVIOURS

In the first set of experiments, the behaviour spaces  $\Pi_i, \Pi_j$  and hypothesis space  $\Pi_j^i$  were restricted to “random” behaviours. Each random behaviour is defined by a sequence of random probability distributions over  $A_j$ . The distributions are created by drawing uniform random numbers from  $(0, 1)$  for each action  $a_j \in A_j$ , and subsequent normalisation so that the values sum up to 1.

Random behaviours are a good baseline for our experiments because they are usually hard to distinguish. This is due to the fact that the entire set  $A_j$  is always in the support of the behaviours, and since they do not react to any past actions. These properties mean that there is little structure in the interaction that can be used to distinguish behaviours.

We simulated 1000 interaction processes, each lasting 10000 time steps. In each process, we randomly sampled behaviours  $\pi_i \in \Pi_i, \pi_j \in \Pi_j$  to control agents  $i$  and  $j$ , respectively. In half of these processes, we used a correct hypothesis  $\pi_j^* = \pi_j$ . In the other half, we sampled a random hypothesis  $\pi_j^* \in \Pi_j^i$  with  $\pi_j^* \neq \pi_j$ . We repeated each set of simulations for  $|A_j| = 2, 10, 20$  (with  $|A_i| = |A_j|$ ) and  $N = 10, 50, 100$  (cf. Section 4.3).

#### 5.1.1 Accuracy & Scalability

Figure 1 shows the average accuracy of our algorithm (for  $N = 50$ ), by which we mean the average percentage of time steps in which the algorithm made correct decisions (i.e. no reject if  $\pi_j^* = \pi_j$ ; reject if  $\pi_j^* \neq \pi_j$ ). The x-axis shows

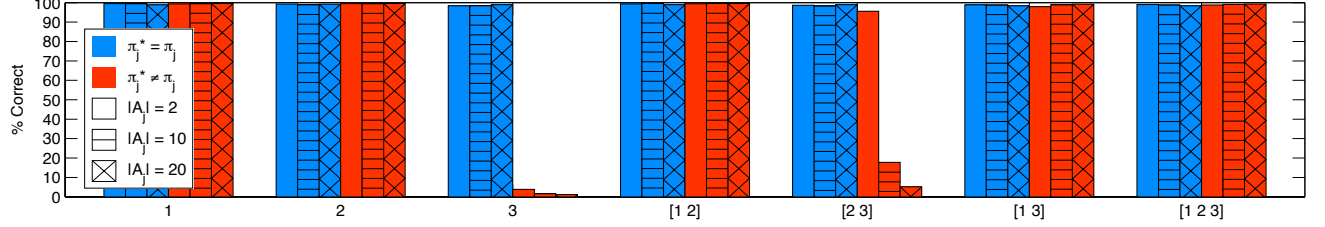


Figure 1: Average accuracy with random behaviours, for  $N = 50$  and  $|A_j| = 2, 10, 20$ . Results averaged over 500 processes with 10000 time steps, for  $\pi_j^* = \pi_j$  and  $\pi_j^* \neq \pi_j$  each. X-axis shows score functions  $z_k$  used in test statistic.

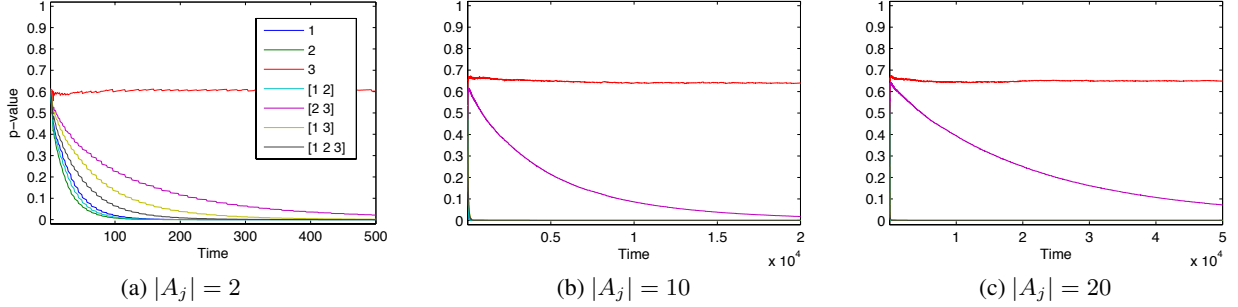


Figure 2: Average  $p$ -values with random behaviours, for  $N = 50$  and  $\pi_j^* \neq \pi_j$  (i.e. hypothesis wrong). Results averaged over 500 processes. Legend shows score functions  $z_k$  used in test statistic.

the combination of score functions used to compute the test statistic (e.g. [1 2] means that we combined  $z_1, z_2$ ).

The results show that our algorithm achieved excellent accuracy, often bordering the 100% mark. They also show that the algorithm scaled well with the number of actions, with no degradation in accuracy. However, there were two exceptions to these observations: Firstly, using  $z_3$  resulted in very poor accuracy for  $\pi_j^* \neq \pi_j$ . Secondly, the combination of  $z_2, z_3$  scaled badly for  $\pi_j^* \neq \pi_j$ .

The reason for both of these exceptions is that  $z_3$  is not a good scoring scheme for random behaviours. The function  $z_3$  quantifies a similarity between the empirical frequency distribution and the averaged hypothesised distributions. For random behaviours (as defined in this work), both of these distributions will converge to the uniform distribution. Thus, under  $z_3$ , any two random behaviours will eventually be the same, which explains the low accuracy for  $\pi_j^* \neq \pi_j$ .

As can be seen in Figure 1, the inadequacy of  $z_3$  is solved when adding any of the other score functions  $z_1, z_2$ . These functions add discriminative information to the test statistic, which technically means that the cardinality  $|\Pi^z|$  in (4) is reduced. However, in the case of  $[z_2, z_3]$ , the converge is substantially slower for higher  $|A_j|$ , meaning that more evidence is needed until  $\pi_j^*$  can be rejected. Figure 2 shows how a higher number of actions affects the average convergence rate of  $p$ -values computed with  $z_2, z_3$ .

In addition to the score functions  $z_k$ , a central aspect for the convergence of  $p$ -values are the corresponding weights

$w_k$  (cf. (5)). As mentioned in Section 4.1, we use uniform weights  $w_k = \frac{1}{K}$ . However, to show that the weighting is not trivial matter, we repeated our experiments with four alternative weighting schemes: Let  $z_k^T = z_k(\hat{\mathbf{a}}_j^T, \pi_j^*) - z_k(\hat{\mathbf{a}}_j^T, \pi_j^*)$  denote the summands in (5). The weighting schemes `truemax`/`truemin` assign  $w_k = 1$  for the first  $k$  that maximises/minimises  $|z_k^T|$ , and 0 otherwise. Similarly, the weighting schemes `max`/`min` assign  $w_k = 1$  for the first  $k$  that maximises/minimises  $z_k^T$ , and 0 otherwise.

Figures 3 and 4 show the results for `truemax` and `truemin`. As can be seen in the figures, `truemax` is very similar to uniform weights while `truemin` improves the convergence for  $[z_2, z_3]$  but compromises elsewhere. The results for `max` and `min` are very similar to those of `truemin` and `truemax`, respectively, hence we omit them.

Finally, we recomputed all accuracies using a more lenient significance level of  $\alpha = 0.05$ . As could be expected, this marginally decreased and increased (i.e. by a few percentage points) the accuracy for  $\pi_j^* = \pi_j$  and  $\pi_j^* \neq \pi_j$ , respectively. Overall, however, the results were very similar to those obtained with  $\alpha = 0.01$ .

### 5.1.2 Sampling Complexity

Recall that  $N$  specifies the number of sampled action vectors  $\tilde{\mathbf{a}}_j^{t,n}$  used to learn the distribution of the test statistic (cf. Section 4.3). In the previous section, we reported results for  $N = 50$ . In this section, we investigate differences in accuracy for  $N = 10, 50, 100$ .

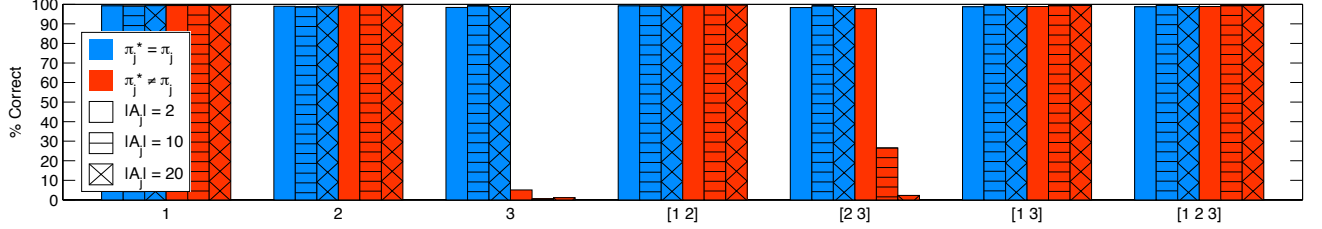


Figure 3: Average accuracy with random behaviours, for  $N = 50$  and  $|A_j| = 2, 10, 20$ . X-axis shows score functions  $z_k$  used in test statistic. Weights  $w_k$  computed using `truemax` weighting.

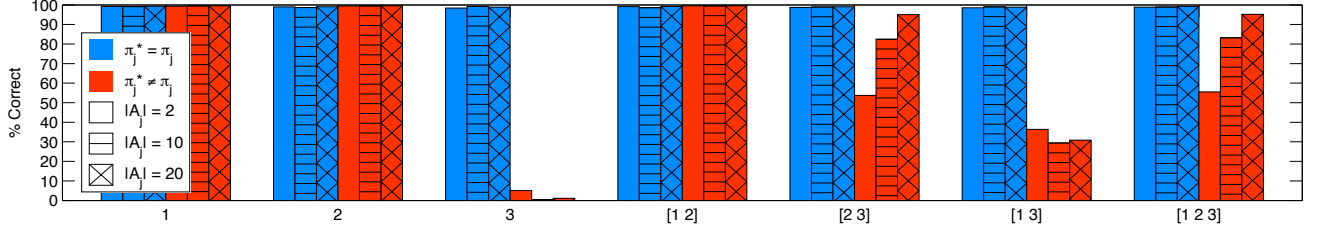


Figure 4: Average accuracy with random behaviours, for  $N = 50$  and  $|A_j| = 2, 10, 20$ . X-axis shows score functions  $z_k$  used in test statistic. Weights  $w_k$  computed using `truemin` weighting.

Figures 5 and 6 show the differences for  $|A_j| = 2, 20$ , respectively. (The figure for  $|A_j| = 10$  was virtually the same as the one for  $|A_j| = 20$ , except with minor improvements in accuracy for the  $[z_2, z_3]$  cluster. Hence, we omit it here.) As can be seen, there were improvements of up to 10% from  $N = 10$  to  $N = 50$ , and no (or very marginal) improvements from  $N = 50$  to  $N = 100$ . This was observed for all  $|A_j| = 2, 10, 20$ , and all constellations of score functions. The fact that  $N = 50$  was sufficient even for  $|A_j| = 20$  is remarkable, since, under random behaviours, there are  $20^t$  possible action vectors to sample at any time  $t$ .

We also compared the learned skew-normal distributions and found that they fitted the data very well. Figures 7 and 8 show the histograms and fitted skew-normal distributions for two example processes after 1000 time steps. In Figure 8, we deliberately chose an example in which the learned distribution was maximally skewed for  $N = 10$ , which is a sign that  $N$  was too small. Nonetheless, in the majority of the processes, the learned distribution was only moderately skewed and our algorithm achieved an average accuracy of 90% even for  $N = 10$ . Moreover, if one wants to avoid maximally skewed distributions, one can simply restrict the parameter space when fitting the skew-normal (specifically, the shape parameter  $\beta$ ; cf. Section 4.3).

The flexibility of the skew-normal distribution was particularly useful in the early stages of the interaction, in which the test statistic typically does not follow a normal distribution. Figure 9 shows the test distribution for an example process after 10 time steps, using  $z_2$  for the test statistic and  $N = 100$  (the histogram was created using  $N = 10000$ ). The learned skew-normal approximated the true test distribution very closely. Note that, in such examples, the normal

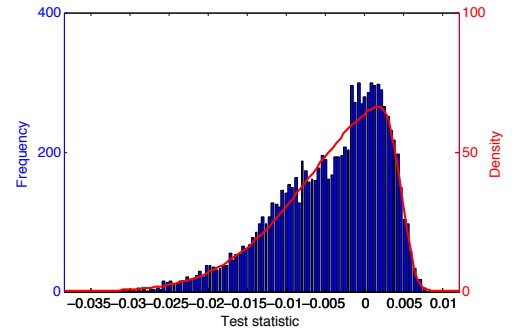


Figure 9: True test distribution for  $z_2$  (histogram) and learned skew-normal distribution (red curve) after 10 time steps, with  $|A_j| = 10$  and  $N = 100$ .

and Student distributions do not produce good fits.

Our implementation of the algorithm performed all calculations as iterative updates (except for the skew-normal fitting). Hence, it used little (fixed) memory and had very low computation times. For example, using all three score functions and  $|A_j| = 20$ ,  $N = 100$ , one cycle in the algorithm (cf. Algorithm 1) took on average less than 1 millisecond without fitting the skew-normal parameters, and less than 10 milliseconds when fitting the skew-normal parameters (using an off-the-shelf Simplex-optimiser with default parameters). The times were measured using Matlab R2014a on a Unix machine with a 2.6 GHz Intel Core i5 processor.

## 5.2 ADAPTIVE BEHAVIOURS

We complemented the “structure-free” interaction of random behaviours by conducting analogous experiments with

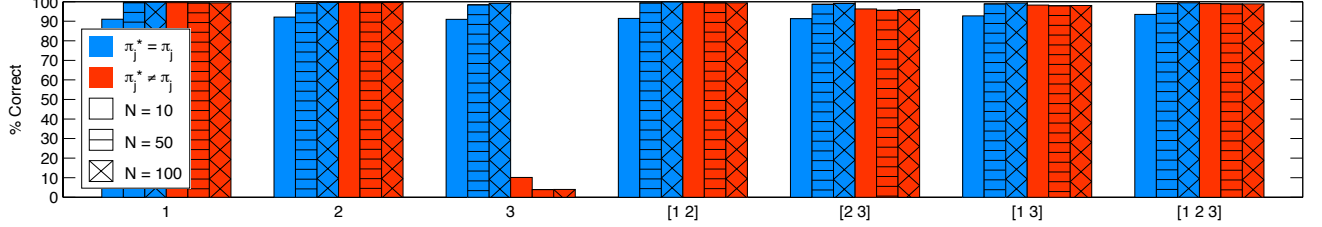


Figure 5: Average accuracy with random behaviours, for  $|A_j| = 2$  and  $N = 10, 50, 100$ . Results averaged over 500 processes with 10000 time steps, for  $\pi_j^* = \pi_j$  and  $\pi_j^* \neq \pi_j$  each. X-axis shows score functions  $z_k$  used in test statistic.

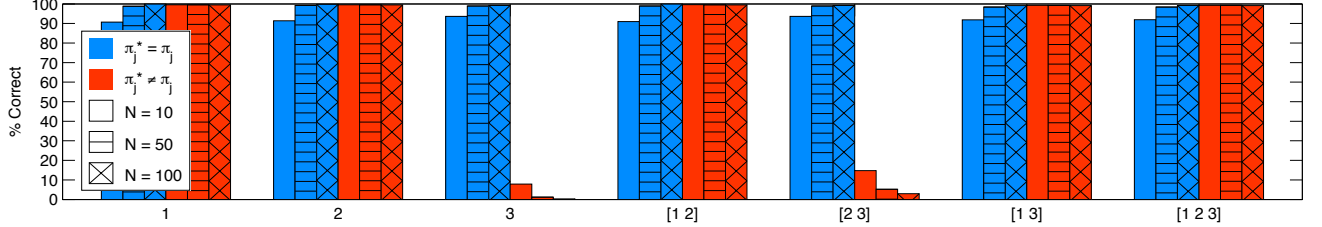


Figure 6: Average accuracy with random behaviours, for  $|A_j| = 20$  and  $N = 10, 50, 100$ . Results averaged over 500 processes with 10000 time steps, for  $\pi_j^* = \pi_j$  and  $\pi_j^* \neq \pi_j$  each. X-axis shows score functions  $z_k$  used in test statistic.

three additional classes of behaviours. Specifically, we used a benchmark framework specified by Albrecht et al. (2015) which consists of 78 distinct  $2 \times 2$  matrix games and three methods to automatically generate sets of behaviours for any given game. The three behaviour classes are Leader-Follower-Trigger Agents (LFT), Co-Evolved Decision Trees (CDT), and Co-Evolved Neural Networks (CNN). These classes cover a broad spectrum of possible behaviours, including fully deterministic (CDT), fully stochastic (CNN), and hybrid (LFT) behaviours. Furthermore, all generated behaviours are *adaptive* to varying degrees (i.e. they adapt their action choices based on the other player’s choices). We refer to Albrecht et al. (2015) for a more detailed description of these classes (we used the same parameter settings).

The following experiments were performed for each behaviour class, using identical randomisation: For each of the 78 games, we simulated 10 interaction processes, each lasting 10000 time steps. For each process, we randomly sampled behaviours  $\pi_i \in \Pi_i, \pi_j \in \Pi_j$  to control agents  $i$  and  $j$ , respectively, where  $\Pi_i, \Pi_j$  (and  $\Pi_j^i$ ) were restricted to the same behaviour class. In half of these processes, we used a correct hypothesis  $\pi_j^* = \pi_j$ , and in the other half, we sampled a random hypothesis  $\pi_j^* \in \Pi_j^i$  with  $\pi_j^* \neq \pi_j$ . As before, we repeated each simulation for  $N = 10, 50, 100$  and all constellations of score functions, but found that there were virtually no differences. Hence, in the following, we report results for  $N = 50$  and the  $[z_1, z_2, z_3]$  cluster.

Figure 10 shows the average accuracy achieved by our algorithm for all three behaviour classes. While the accuracy for  $\pi_j^* = \pi_j$  was generally good, the accuracy for  $\pi_j^* \neq \pi_j$  was mixed. Note that this was not merely due to the fact that the score functions were imperfect (cf. Section 4.1), since we

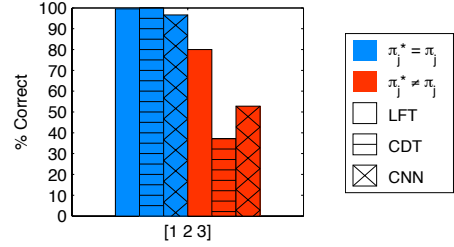


Figure 10: Average accuracy for behaviour classes LFT, CDT, CNN ( $N = 50$ ).  $\Pi_i$  and  $\Pi_j$  restricted to same class.

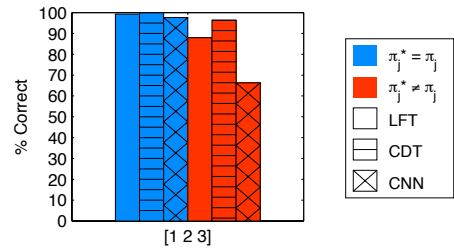


Figure 11: Average accuracy for behaviour classes LFT, CDT, CNN ( $N = 50$ ).  $\Pi_i$  set to random behaviours.

obtained the same results for all combinations. Rather, this reveals an inherent limitation of our approach, which is that *we do not actively probe aspects of the hypothesis  $\pi_j^*$* . In other words, our algorithm performs statistical hypothesis tests based only on evidence that was generated by  $\pi_i$ .

To illustrate this, it is useful to consider the tree structure of behaviours in the CDT class. Each node in a tree  $\pi_j$  corresponds to a past action taken by  $\pi_i$ . Depending on how  $\pi_i$  chooses actions, we may only ever see a subset of the

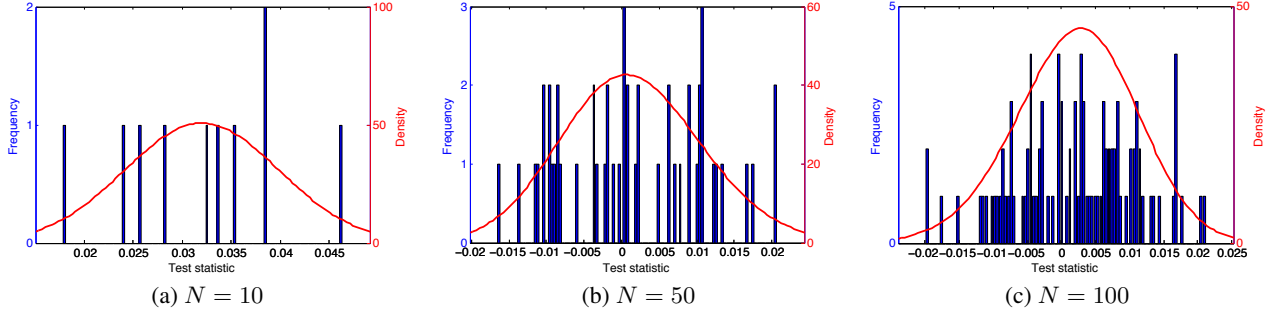


Figure 7: Example histograms and fitted skew-normal distributions (red curve) after 1000 time steps, for random behaviours with  $|A_j| = 10$  and  $N = 10, 50, 100$ . Using score function  $z_1$  in test statistic.

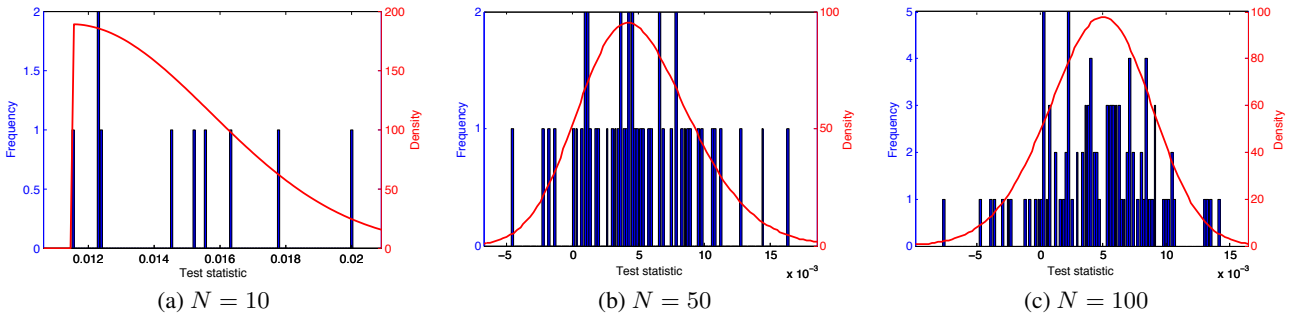


Figure 8: Example histograms and fitted skew-normal distributions (red curve) after 1000 time steps, for random behaviours with  $|A_j| = 10$  and  $N = 10, 50, 100$ . Using score functions  $z_1, z_2, z_3$  in test statistic.

entire tree that defines  $\pi_j$ . However, if our hypothesis  $\pi_j^*$  differs from  $\pi_j$  only in the unseen aspects of  $\pi_j$ , then there is no way for our algorithm to differentiate the two. Hence the asymmetry in accuracy for  $\pi_j^* = \pi_j$  and  $\pi_j^* \neq \pi_j$ . Note that this problem did not occur in random behaviours because, there, all aspects are eventually visible.

Following this observation, we repeated the same experiments but restricted  $\Pi_i$  to random behaviours, with the goal of exploring  $\pi_j^*$  more thoroughly. As shown in Figure 11, this led to significant improvements in accuracy, especially for the CDT class. Nonetheless, choosing actions purely randomly may not be a sufficient probing strategy, hence the accuracy for CNN was still relatively low. For CNN, this was further complicated by the fact that two neural networks  $\pi_j, \pi'_j$  may formally be different ( $\pi_j \neq \pi'_j$ ) but have essentially the same action probabilities (with extremely small differences). Hence, in such cases, we would require much more evidence to distinguish the behaviours.

## 6 CONCLUSION

We hold the view that if an intelligent agent is to interact effectively with other agents whose behaviours are unknown, it will have to hypothesise what these agents might be doing *and* contemplate the truth of its hypotheses, such that appropriate measures can be taken if they are deemed false. In this spirit, we presented a novel algorithm which decides this

question in the form of a frequentist hypothesis test. The algorithm can incorporate multiple statistical criteria into the test statistic and learns the test distribution during the interaction process, with asymptotic correctness guarantees. We presented results from a comprehensive set of experiments, showing that our algorithm achieved high accuracy and scalability at low computational costs.

There are several directions for future work: To bring some structure into the space of score functions, we introduced the concepts of consistency and perfection as minimal and ideal properties. However, more research is needed to understand precisely what properties a useful score function should satisfy, and whether the concept of perfection is feasible or even necessary in the general case. Furthermore, we used uniform weights to combine the computed scores into a test statistic, and we also experimented with alternative weighting schemes to show that the weighting can have a substantial effect on convergence rates. However, further research is required to understand the effect of weights on decision quality and convergence.

Finally, in this work, we assumed that the behaviour of the other agent ( $j$ ) could be described as a function of the information available to our agent ( $i$ ). An important extension would be to also account for information that cannot be deterministically derived from our observations, especially in the context of robotics where observations are often described as random variables.

## References

- S.V. Albrecht and S. Ramamoorthy. On convergence and optimality of best-response learning with policy types in multiagent systems. In *Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence*, pages 12–21, 2014.
- S.V. Albrecht, J.W. Crandall, and S. Ramamoorthy. An empirical study on the practical impact of prior beliefs over policy types. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, pages 1988–1994, 2015.
- A. Azzalini. A class of distributions which includes the normal ones. *Scandinavian Journal of Statistics*, 12:171–178, 1985.
- I.V. Basawa and D.J. Scott. Efficient tests for stochastic processes. *Sankhyā: The Indian Journal of Statistics, Series A*, pages 21–31, 1977.
- M.J. Bayarri and J.O. Berger. P values for composite null models. *Journal of the American Statistical Association*, 95(452):1127–1142, 2000.
- J.O. Berger and T. Sellke. Testing a point null hypothesis: the irreconcilability of  $p$  values and evidence (with discussion). *Journal of the American Statistical Association*, 82:112–122, 1987.
- G.E.P. Box. Sampling and Bayes’ inference in scientific modelling and robustness. *Journal of the Royal Statistical Society. Series A (General)*, pages 383–430, 1980.
- G.W. Brown. Iterative solution of games by fictitious play. *Activity Analysis of Production and Allocation*, 13(1):374–376, 1951.
- S. Carberry. Techniques for plan recognition. *User Modeling and User-Adapted Interaction*, 11(1-2):31–48, 2001.
- D. Carmel and S. Markovitch. Exploration strategies for model-based learning in multi-agent systems: Exploration strategies. *Autonomous Agents and Multi-Agent Systems*, 2(2):141–172, 1999.
- E. Charniak and R.P. Goldman. A Bayesian model of plan recognition. *Artificial Intelligence*, 64(1):53–79, 1993.
- E.M. Clarke, O. Grumberg, and D.A. Peled. *Model Checking*. MIT Press, 1999.
- V. Conitzer and T. Sandholm. AWESOME: A general multiagent learning algorithm that converges in self-play and learns a best response against stationary opponents. *Machine Learning*, 67(1-2):23–43, 2007.
- D.R. Cox. The role of significance tests (with discussion). *Scandinavian Journal of Statistics*, 4:49–70, 1977.
- H. Fischer. *A History of the Central Limit Theorem: From Classical to Modern Probability Theory*. Springer Science & Business Media, 2010.
- R.A. Fisher. *The Design of Experiments*. Oliver & Boyd, 1935.
- D.P. Foster and H.P. Young. Learning, hypothesis testing, and Nash equilibrium. *Games and Economic Behavior*, 45(1):73–96, 2003.
- A. Gelman and C.R. Shalizi. Philosophy and the practice of Bayesian statistics. *British Journal of Mathematical and Statistical Psychology*, 66(1):8–38, 2013.
- I. Gilboa and D. Schmeidler. *A Theory of Case-Based Decisions*. Cambridge University Press, 2001.
- P.J. Gmytrasiewicz and P. Doshi. A framework for sequential planning in multiagent settings. *Journal of Artificial Intelligence Research*, 24(1):49–79, 2005.
- K.G. Larsen and A. Skou. Bisimulation through probabilistic testing. *Information and Computation*, 94(1):1–28, 1991.
- X.-L. Meng. Posterior predictive p-values. *The Annals of Statistics*, pages 1142–1160, 1994.
- A. O’Hagan and T. Leonard. Bayes estimation subject to uncertainty about parameter constraints. *Biometrika*, 63(1):201–203, 1976.
- D.B. Rubin. Bayesianly justifiable and relevant frequency calculations for the applied statistician. *The Annals of Statistics*, 12(4):1151–1172, 1984.
- D. Ryabko and B. Ryabko. On hypotheses testing for ergodic processes. In *Proceedings of IEEE Information Theory Workshop*, pages 281–283, 2008.
- A. Vehtari and J. Ojanen. A survey of Bayesian predictive methods for model assessment, selection and comparison. *Statistics Surveys*, 6:142–228, 2012.
- Y. Yue, Y. Gao, O. Chapelle, Y. Zhang, and T. Joachims. Learning more powerful test statistics for click-based retrieval evaluation. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 507–514, 2010.

---

# Disciplined Convex Stochastic Programming: A New Framework for Stochastic Optimization

---

**Alnur Ali**  
Machine Learning Dept.  
Carnegie Mellon University  
alnurali@cmu.edu

**J. Zico Kolter**  
School of Computer Science  
Carnegie Mellon University  
zkolter@cs.cmu.edu

**Steven Diamond**  
Dept. of Computer Science  
Stanford University  
stevend2@stanford.edu

**Stephen Boyd**  
Dept. of Electrical Engineering  
Stanford University  
boyd@stanford.edu

## Abstract

We introduce *disciplined convex stochastic programming* (DCSP), a modeling framework that can significantly lower the barrier for modelers to specify and solve convex stochastic optimization problems, by allowing modelers to naturally express a wide variety of convex *stochastic programs* in a manner that reflects their underlying mathematical representation. DCSP allows modelers to express expectations of arbitrary expressions, *partial optimizations*, and *chance constraints* across a wide variety of convex optimization problem families (*e.g.*, linear, quadratic, second order cone, and semidefinite programs). We illustrate DCSP’s expressivity through a number of sample implementations of problems drawn from the operations research, finance, and machine learning literatures.

## 1 INTRODUCTION

We introduce *disciplined convex stochastic programming* (DCSP), a modeling framework for specifying and solving convex *stochastic programs*: convex optimization problems that include random variables. DCSP builds on principles from stochastic optimization and convex analysis to allow modelers to naturally express a wide variety of stochastic programs in a manner that reflects their underlying mathematical representation. At a high level, DCSP enables modelers to specify — in a straightforward way — and solve convex optimization problems that include (1) expectations of arbitrary expressions, (2) *partial optimizations*, optimizations over (only) a subset of the optimization variables, which additionally pave the way for the specification of *multi-stage* stochastic programs (Sec. 2.1), and (3) *chance constraints*, constraints that are required to hold with high probability — these three building blocks can be used to express a wide variety of stochastic optimization problems.

Concurrently with this paper, we also make available an open source Python implementation of DCSP, which we refer to as `cvxstoc`<sup>1</sup>, that allows modelers to write and solve stochastic programs — we present a variety of examples of using `cvxstoc` to model stochastic optimization problems, drawn from the operations research, finance, and machine learning literatures, in Sec. 4.

**Related work** Although other frameworks for stochastic programming do exist ([24, 20, 11], and in Python mainly [26]), they often require significant effort from the modeler to manipulate the optimization problem into an amenable form, support a limited number of stochastic programming constructs (*e.g.*, [11] only supports chance constraints with uncertainty sets), and cannot express certain families of convex optimization problems; indeed, checking the convexity of and solving (convex) optimization problems in general is challenging. DCSP builds on (and extends) *disciplined convex programming* (DCP) [10], a recently introduced framework that makes it natural for modelers to express convex optimization problems, and additionally automates the tasks of verifying the convexity of these problems and translating them into conic form (see, *e.g.*, [8]). This means that DCSP can be used to express and solve a wide variety of stochastic convex optimization problems, including linear, quadratic, second order cone, and semidefinite programs. Probabilistic programming languages (*e.g.*, [9, 16, 13]) offer an alternative approach, but tend to focus on inference problems, and may not contain the features to capture traditional stochastic programming problem formulations; in contrast, convex modeling can be attractive because local solutions are global solutions, efficient solvers exist, and guarantees can often be obtained on the optimality of a solution obtained by a solver.

This paper is structured as follows. In Sec. 2, we review background on stochastic programming, DCP, and `cvxpy` (an open source Python implementation of DCP). In Sec. 3, we describe DCSP and `cvxstoc`’s syntax. In

---

<sup>1</sup>`cvxstoc` is available as an extension of the `cvxpy` Python package [7]: see <http://www.cvxpy.org>.



Sec. 4, we present a number of examples that illustrate our framework.

## 2 BACKGROUND

### 2.1 STOCHASTIC PROGRAMMING

A convex optimization problem has the form

$$\begin{aligned} & \text{minimize} && f_0(x) \\ & \text{subject to} && f_i(x) \leq 0, \quad i = 1, \dots, m, \\ & && h_i(x) = 0, \quad i = 1, \dots, p, \end{aligned}$$

where  $x \in \mathbf{R}^n$  is the optimization variable,  $f_0 : \mathbf{R}^n \rightarrow \mathbf{R}$  is a convex objective function,  $f_i : \mathbf{R}^n \rightarrow \mathbf{R}$ ,  $i = 1, \dots, m$  are convex inequality constraint functions, and  $h_i : \mathbf{R}^n \rightarrow \mathbf{R}$ ,  $i = 1, \dots, p$  are affine equality constraint functions.

A convex stochastic program has the form

$$\begin{aligned} & \text{minimize} && \mathbf{E} f_0(x, \omega) \\ & \text{subject to} && \mathbf{E} f_i(x, \omega) \leq 0, \quad i = 1, \dots, m \\ & && h_i(x) = 0, \quad i = 1, \dots, p, \end{aligned} \quad (1)$$

where  $f_i : \mathbf{R}^n \times \mathbf{R}^q \rightarrow \mathbf{R}$ ,  $i = 0, \dots, m$  are convex functions in  $x$  for each value of a random variable  $\omega \in \mathbf{R}^q$ , and  $h_i : \mathbf{R}^n \rightarrow \mathbf{R}$ ,  $i = 1, \dots, p$  are (deterministic) affine functions; since expectations preserve convexity, the objective and inequality constraint functions in (1) are (also) convex in  $x$ , making (1) a convex optimization problem.

**Two-stage stochastic programs** An important special case of (1) is a so-called *two-stage* stochastic program (also referred to as an optimization problem with *recourse*) [6]:

$$\begin{aligned} & \text{minimize} && f_0(x) + \mathbf{E} Q(x, \omega) \\ & \text{subject to} && f_i(x) \leq 0, \quad i = 1, \dots, m, \\ & && h_i(x) = 0, \quad i = 1, \dots, p, \end{aligned} \quad (2)$$

$$\text{where } Q(x, \omega) = \inf_y \{ \phi_0(x, y, \omega) : \phi_i(x, y, \omega) \leq 0, \psi_j(x, y) = 0, i = 1, \dots, s, j = 1, \dots, w \}$$

is the *second stage* problem,  $y \in \mathbf{R}^r$  is the second stage optimization variable,  $\phi_0 : \mathbf{R}^n \times \mathbf{R}^r \times \mathbf{R}^q \rightarrow \mathbf{R}$  is the second stage objective function, and is convex in  $(x, y)$  for each value of  $\omega$ ,  $\phi_i : \mathbf{R}^n \times \mathbf{R}^r \times \mathbf{R}^q \rightarrow \mathbf{R}$ ,  $i = 1, \dots, s$  are the second stage inequality constraint functions, also convex in  $(x, y)$  for each value of  $\omega$ , and  $\psi_i : \mathbf{R}^n \times \mathbf{R}^r \rightarrow \mathbf{R}$ ,  $i = 1, \dots, w$  are the second stage equality constraint functions, and are affine in  $(x, y)$ . That is,  $Q$  is itself the optimal value of another convex optimization problem, and is convex in  $x$  for each value of  $\omega$ .

Two-stage stochastic programs model the uncertain consequences (in the second stage) of here-and-now decision-making (in the first stage): *e.g.*, in a finance application, we may wish to decide which assets to purchase now, while (also) factoring in how the asset prices might fluctuate later.

**Chance-constrained problems** A *chance constraint* [5] is a constraint on the variable  $x$  of the form

$$\text{Prob}(f(x, \omega) \leq 0) \geq \eta,$$

where  $f$  is convex in  $x$  for each value of  $\omega$ , and  $\eta$  is typically a large probability (*e.g.*, 0.95); a *chance-constrained problem* is an optimization problem with one or more chance constraints. Chance constraints are typically non-convex, although effective convex approximations exist (see Sec. 3.3).

### 2.2 DISCIPLINED CONVEX PROGRAMMING

Disciplined convex programming (DCP) is a recently introduced modeling framework for specifying and solving convex optimization problems [10]. In a nutshell, DCP consists of a library of convex atomic functions, and a *convex rule-set* that prescribes how these atomic functions may be composed to express (more complex) convex optimization problems.

**Convex rule-set** Verifying the convexity of arbitrary expressions is challenging; DCP checks convexity using Thm. 2.1, which is equivalent to enforcing a set of rules.

**Theorem 2.1** ([10]). *Suppose  $f = h(g_1(x), \dots, g_k(x))$ , where  $h : \mathbf{R}^k \rightarrow \mathbf{R}$  is convex and  $g_i : \mathbf{R}^n \rightarrow \mathbf{R}$ ,  $i = 1, \dots, k$ , and one of the following holds for each  $i = 1, \dots, k$ :*

- $g_i$  is convex and  $h$  is nondecreasing in argument  $i$
- $g_i$  is concave and  $h$  is nonincreasing in argument  $i$
- $g_i$  is affine.

Then  $f$  is convex<sup>2</sup>.

Thm. 2.1 permits a wide variety of convex expressions: for example, the maximum eigenvalue of a symmetric matrix,  $\lambda_{\max}(2X - 4I)$ , where  $X \in \mathbf{S}^n$ , is recognized as convex. (On the other hand, as an example of a limitation of the rule-set, the expression  $(\sum_{i=1}^n x_i^2)^{1/2}$ , where  $x \in \mathbf{R}^n$ , is *not* recognized as convex, although it *is* recognized as convex once reformulated as  $\|x\|_2$ .)

**Library of atoms** Atomic convex functions<sup>3</sup> are specified in DCP in their epigraph form: for example, the (convex) function  $f(x) = \|x\|_1$  is specified as

$$\begin{aligned} & \text{minimize} && 1^T t \\ & \text{subject to} && -t \preceq x \preceq t, \end{aligned} \quad (3)$$

<sup>2</sup>A similar result holds for concave functions.

<sup>3</sup>See <http://www.cvxpy.org/en/latest/tutorial/functions/index.html> for a list of the convex atoms available in cvxpy.

where  $x, t \in \mathbf{R}^n$ . Thus, whenever a modeler writes the atom  $f(x) = \|x\|_1$ , DCP internally replaces it with (3), introduces the variable  $t$ , and can subsequently optimize over  $(x, t)$ .

**Disciplined convex programming** DCP certifies a problem’s convexity by constructing an abstract syntax tree for the objective and constraint functions, with atoms as internal nodes, and variables and constants as leaves, and then applying Thm. 2.1 recursively [10, 22].

### 2.3 cvxpy

cvxpy [7] is an open source Python DCP implementation; we briefly describe its syntax next.

Variables are declared simply in cvxpy as follows:

```
x = Variable()
x = NonNegative()
X = Semidefinite(n)
```

The first line declares  $x$  to be a variable in  $\mathbf{R}$ , the second declares  $x$  to be a variable in the nonnegative orthant  $\mathbf{R}_+$ , and the third declares  $X$  to be a  $(n \times n)$  matrix variable in the positive semidefinite cone  $\mathbf{S}_+^n$ .

Convex expressions are specified by composing convex atoms; for example, the log loss  $\sum_{i=1}^m \log(1 + \exp(-y_i(w^T x_i + b)))$  can be specified by using the simpler `log_sum_exp` atom as follows:

```
expr = [log_sum_exp(vstack(0, -y[i]*(w.T*x[i]+b)))
        for i in range(m)]
```

An objective is specified by instantiating a *sense* (i.e., `Minimize` or `Maximize`) with an expression:

```
obj = x.T*c
Minimize(obj)
```

Constraints are specified by forming a list of expressions:

```
constrs = [x >= 0, x.T*numpy.ones((n,1)) == 1, ...]
```

A convex optimization problem, then, is specified by instantiating a `Problem` with an objective and a list of constraints:

```
prob = Problem(Minimize(obj), constrs)
prob.solve()
```

The last line solves the optimization problem.

## 3 DISCIPLINED CONVEX STOCHASTIC PROGRAMMING

In this section we present the chief methodological contribution of the paper: the disciplined convex stochastic programming (DCSP) framework, along with an overview of

its implementation in the `cvxstoc` Python package. In a nutshell, DCSP consists of the addition of three operations to the disciplined convex programming (DCP) framework, which can be used to express a wide variety of convex stochastic programs: the ability to (1) compute (approximations to) expectations of arbitrary expressions, (2) handle *partial optimization*, and (3) compute (approximations to) chance constraints.

### 3.1 RANDOM VARIABLES AND EXPECTATIONS

**Random variables** The most fundamental operations in stochastic programs, and hence in DCSP, are the ability to specify random variables, and compute (approximations to) expectations of arbitrary expressions containing these random variables. As in Sec. 2, DCSP assumes that all expressions in a stochastic program are convex in the optimization variable(s) for each value of the random variable(s) — thus, from the point of view of DCSP, random variables do not affect the convexity of their parent expressions and can be regarded as equivalent to constants, thereby requiring no additions to the DCP convex rule-set. (Practically speaking, DCSP permits the specification of a variety of random variables; see Sec. 3.4.)

**Expectations** DCSP computes (approximations to) expectations of arbitrary expressions using simple Monte Carlo evaluation, i.e.,

$$\mathbf{E} f(x, \omega) \approx (1/N) \sum_{i=1}^N f(x, \omega_i),$$

where  $f$  is (again) assumed to be convex in the optimization variable  $x$  for each value of the random variable  $\omega$ , and  $\omega_i, i = 1, \dots, N$  are samples of  $\omega$ ; this approximation is referred to as the *sample average approximation* (SAA) in the stochastic programming literature, and methods that use it are often referred to as *scenario-based* methods. By the DCP rule-set, the nonnegative weighted sum of convex functions is a convex function; thus, the expectation operator applied to an expression that is convex in  $x$  returns an expression that is (also) convex in  $x$ .

The SAA is, of course, a very simple method for approximating an expectation, and much more involved methods for solving stochastic programs exist, but the clear advantage of this method is its simplicity: any random variable can be included in a stochastic program as long as we are able to draw samples of it. If  $\omega$  is a discrete random variable, then DCSP calculates its expectation exactly; otherwise, DCSP draws samples using Markov chain Monte Carlo (MCMC) methods<sup>4</sup> [15].

In the case of unconstrained stochastic programs, the SAA objective value is (naturally) an unbiased estimator of the

<sup>4</sup>We implement MCMC by leveraging the `PyMC` Python package [15].

true objective value,  $\mathbf{E} f_0(x, \omega)$ , with variance  $\propto 1/N$ , and an asymptotically normal distribution [21, chap. 5]. Thm. 3.1 additionally tells us that (roughly) both the optimal value and optimal set of a SAA converge almost surely to the optimal value and optimal set of the true problem.

**Theorem 3.1** ([21, Thm. 5.3]). *Define  $\hat{p}_N^*$ ,  $\hat{S}_N$  and  $p^*$ ,  $S$  as the optimal value and optimal set of a SAA with  $N$  samples and of the true problem, respectively, and let  $K \subset \mathbf{R}^n$  be a compact set. Suppose (a)  $S \subseteq K$  is nonempty, (b)  $\hat{S}_N \subseteq K$  is nonempty a.s., (c)  $f_0$  is finite and continuous on  $K$ , and (d)  $(1/N) \sum_{i=1}^N f_0(x, \omega_i) \xrightarrow{\text{a.s.}} f(x)$  (uniformly) for  $x \in K$ . Then  $\hat{p}_N^* \xrightarrow{\text{a.s.}} p^*$  and  $\sup_{x \in \hat{S}_N} \inf_{y \in S} \|x - y\|_2 \xrightarrow{\text{a.s.}} 0$ .*

$\hat{p}_N^*$  is also a downward biased estimator of  $p^*$ , although its bias decreases with  $N$  [21, Prop. 5.6]. In Sec. 3.4, we empirically investigate the quality of the SAA.

### 3.2 PARTIAL OPTIMIZATION

DCSP adds a new partial optimization atom to the DCP atom library, allowing modelers to express partial optimizations, *i.e.*, optimizations over (only) a subset of the optimization variables; this atom also forms the basis for specifying two-stage stochastic programs.

We start with the observation that partial optimization is a convex operation (see, *e.g.*, [4, page 87]): *i.e.*, if  $f$  is convex in  $(x, y)$  and  $C$  is a nonempty convex set, then

$$g(x) := \inf_y \{f(x, y) : (x, y) \in C\},$$

is convex in  $x$ .

Accordingly, DCSP specifies a new partial optimization atom that takes as input a convex optimization problem and returns (the epigraph form for) another convex atom, which complies with the DCP prescription for specifying atoms — this means that modelers can use partial optimizations in stochastic programs as they would other atoms. In particular, two-stage stochastic programs, *i.e.*, (2), can be naturally expressed using this atom; furthermore, the second stage optimization problem  $Q$  need not be in standard form (as required by other frameworks).

### 3.3 CHANCE CONSTRAINTS

DCSP computes conservative approximations to chance constraints, as they are typically nonconvex<sup>5</sup>; in particular, DCSP replaces

$$\mathbf{Prob}(f(x, \omega) \geq 0) \leq 1 - \eta, \quad (4)$$

<sup>5</sup>One notable exception is chance constraints involving affine functions of normal random variables, which can be expressed as a second order cone constraint (see, *e.g.*, [4, page 157]). However, we favor the approximate approach described in this section because it is substantially more general, and applies to any class of random variables.

with a convex upper bound derived as follows [3]. Suppose  $\phi : \mathbf{R} \rightarrow \mathbf{R}_+$  is a nonnegative, increasing convex function with  $\phi(0) = 1$ ; then  $\phi(z) \geq 1(z \geq 0)$ , where  $1(z \geq 0)$  equals 1 if  $z \geq 0$  and 0 otherwise, and so  $\phi(z/\alpha) \geq 1(z \geq 0)$ , for some variable  $\alpha \in \mathbf{R}_{++}$ . Thus

$$\mathbf{E} \phi(f(x, \omega)/\alpha) \geq \mathbf{Prob}(f(x, \omega) \geq 0),$$

and so

$$\begin{aligned} \alpha \mathbf{E} \phi(f(x, \omega)/\alpha) &\leq \alpha(1 - \eta) \\ \implies \mathbf{Prob}(f(x, \omega) \geq 0) &\leq 1 - \eta, \end{aligned} \quad (5)$$

*i.e.*, (5) is a conservative approximation to (4). Note that (5) is convex in  $(x, \alpha)$ : it is the perspective of the expectation of a convex increasing function,  $\phi$ , of a convex function,  $f$ .<sup>6</sup>

In (5),  $\alpha$  can be interpreted as modulating the “steepness” of the approximation; several choices of  $\phi$  are possible, and are analogous, *e.g.*, to different approximations to the zero-one loss common in machine learning. DCSP uses  $\phi(z) = \max\{0, z + 1\}$ , which roughly corresponds to a Markov-inequality type bound<sup>7</sup> on (4), and can also be interpreted as the *conditional value-at-risk* of  $f(x, \omega)$  [19]. Prop. 3.2 also tells us that this is the tightest possible choice of  $\phi$ .

Practically speaking, DCSP approximates (5) with its SAA (at which point all the benefits of DCP readily apply), then optimizes over  $(x, \alpha)$  to obtain the tightest possible bound; in Sec. 4.3, we empirically investigate the quality of these approximations.

**Proposition 3.2** ([14]). *Suppose  $\phi : \mathbf{R} \rightarrow \mathbf{R}_+$  is a nonnegative, increasing convex function and  $\phi(z) \geq 1(z \geq 0)$ ; then  $\exists \alpha \in \mathbf{R}_+$  such that  $\mathbf{E}(f(x, \omega)/\alpha + 1)_+ \leq \mathbf{E} \phi(f(x, \omega))$ .*

### 3.4 cvxstoc

Next, we briefly detail the syntax of `cvxstoc`; `cvxstoc` builds on `cvxpy`, and thus much of the usage is similar.

Random variables are specified simply in `cvxstoc` as follows:

```
omega = RandomVariableFactory().create_normal_rv(0,1)
```

Here, `omega` is a standard normal random variable. `cvxstoc` includes a `RandomVariableFactory` object to simplify the specification of common random variables; see Sec. 4 for examples of the specification of other random variables.

Expectations are specified by applying the `expectation` atom:

<sup>6</sup>Alternatively, the modeler can fix  $\alpha$ , in which case the bound is convex in  $(x, \eta)$  if desired.

<sup>7</sup>Alternatively, one could take  $\phi(z) = \exp z$ , which would be analogous to a Chernoff-type inequality.

```
result = expectation(exp(omega*x), m)
```

Here,  $\exp(\omega x)$  is the expression we wish to compute the expectation of, and  $m$  is the number of Monte Carlo samples to use when constructing the SAA.

Partial optimizations are specified by applying the `partial_optimize` atom:

```
atom = partial_optimize(prob, [y], [x])
```

The first argument here is a `Problem`, the second is a list of variables to optimize over, and the third is a list of variables to *not* optimize over.

Two-stage stochastic programs can, in turn, be specified as follows:

```
Q = partial_optimize(prob2, [y], [x])
probl = Problem(Minimize(f0 + expectation(Q(x),m)),
               constrs)
```

Here, `prob2` is the second stage problem, `y` is the second stage variable, `x` is the first stage variable, and `probl` is the first stage problem. Multi-stage stochastic programs can be specified by iterating this construction:

```
Q2 = partial_optimize(prob3, [z], [x,y])
prob2 = Problem(Minimize(phi0 + expectation(Q2(y),m)),
               constrs2)
Q1 = partial_optimize(prob2, [y], [x])
probl = Problem(Minimize(f0 + expectation(Q1(x),m)),
               constrs1)
```

Chance constraints are specified by instantiating the `prob` class and chaining it with an inequality:

```
prob(constr >= 0, m) <= 1-eta
```

Here, `constr` is a (stochastic) constraint, and `m` is the number of Monte Carlo samples to use.

### 3.4.1 Quality of the sample average approximation

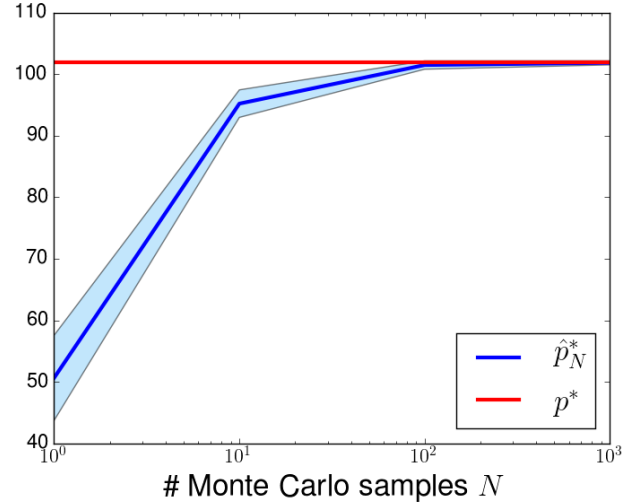
Here, we investigate the quality of the SAA employed by `cvxstoc` in the special case of a (unconstrained) least squares problem

$$\underset{x}{\text{minimize}} \quad \mathbf{E} \|Ax - b\|_2^2, \quad (6)$$

where the entries of  $A \in \mathbf{R}^{m \times n} \sim \text{Normal}(\mu_1, \sigma_1^2)$ ,  $b \in \mathbf{R}^m \sim \text{Normal}(\mu_2, \sigma_2^2)$ , and  $x \in \mathbf{R}^n$ , in which case the objective has the analytic form

$$x^T \mathbf{E} A^T A x - 2 \mathbf{E} b^T A x + \mathbf{E} b^T b,$$

assuming we know the second moments of  $(A, b)$ . Fig. 1 plots the optimal value of the true problem (6) and a SAA to (6): we see that the SAA obtains reasonable accuracy after roughly 100 Monte Carlo samples.



**Figure 1:** The optimal values of the stochastic least squares problem (6) (red) and a SAA to (6) (blue) with 95% confidence intervals (light blue) vs. the number of Monte Carlo samples; in this case,  $m = 100, n = 50$ , although similar results hold across a variety of problem sizes.

## 4 EXAMPLES

At this point, we switch gears slightly and present several examples of stochastic programs along with their corresponding `cvxstoc` implementations<sup>8</sup>; the majority of these applications are well established or previously known, though we also include some formulations that are novel, to the best of our knowledge (namely, the precise formulation of the stochastic optimal power flow problem in Sec. 4.4, and the budgeted learning of a classifier in a cascade problem in Sec. 4.6).

### 4.1 YIELD-CONSTRAINED COST MINIMIZATION

We begin with a simple example from the operations research literature (see, *e.g.*, [4, page 107]). Consider the (general) problem of choosing the parameters  $x \in \mathbf{R}^n$  governing a manufacturing process so that our cost  $c^T x$ , where  $c \in \mathbf{R}^n$ , is minimized, while the parameters lie in a set of allowable values  $S$ ; we can model noise in the manufacturing process by expressing this constraint as  $\text{Prob}(x + \omega \in S) \geq \eta$ , where  $\omega \in \mathbf{R}^n$  is a random vector and  $\eta$  is a large probability (*e.g.*, 0.95), which is referred to as an  $\eta$ -yield constraint. Thus, we have the optimization problem

$$\begin{aligned} & \underset{x}{\text{minimize}} && c^T x \\ & \text{subject to} && \text{Prob}(x + \omega \in S) \geq \eta. \end{aligned}$$

<sup>8</sup>Due to space constraints, we present one of these examples in the supplementary material.

Note that if the distribution over  $\omega$  is log-concave and  $S$  is a convex set, then this constraint is convex in  $x$ . We can directly express the yield-constrained cost minimization problem using `cvxstoc`; an implementation is given in Listing 1 ( $S$  is taken to be an ellipsoid).

```
# Create problem data
n = 10
c = numpy.random.randn(n)
P, q, r = numpy.eye(n), numpy.random.randn(n), numpy.
    random.randn()
mu, Sigma = numpy.zeros(n), 0.1*numpy.eye(n)
omega = RandomVariableFactory().create_normal_rv(mu,
    Sigma)
m, eta = 100, 0.95

# Create and solve optimization problem
x = Variable(n)
yield_constr = prob(quad_form(x+omega,P)
    + (x+omega).T*q + r >= 0, m) <= 1-eta
p = Problem(Minimize(x.T*c), [yield_constr])
p.solve()
```

**Listing 1:** A `cvxstoc` implementation of the yield-constrained cost minimization problem.

## 4.2 THE NEWS VENDOR PROBLEM

The news vendor problem is a classic problem in the stochastic programming literature (see, e.g., [2, page 15]); in this problem, a vendor must decide how much newspaper to stock, so that profit is maximized while backorder and return fees (due to excess or insufficient demand, respectively) are minimized, in the face of uncertain demand.

Our optimization variables are the number of units of stocked newspaper  $x \in \mathbf{R}_+$ , the number of units purchased by customers  $y_1 \in \mathbf{R}_+$ , and the number of unpurchased (surplus) units that must be returned by the vendor  $y_2 \in \mathbf{R}_+$ . Our problem data are  $b, s, r \in \mathbf{R}_+$ , which denote the price to stock, sell, and return a unit of newspaper, respectively. Lastly, we let the random variable  $d \sim \text{Categorical}$  model the uncertain (newspaper) demand.

We can pose the news vendor problem as the following two-stage stochastic program:

$$\begin{aligned} & \underset{x}{\text{minimize}} && bx + \mathbf{E} Q(x) \\ & \text{subject to} && 0 \leq x \leq u, \end{aligned}$$

where  $Q(x) = \min_{y_1, y_2} -(sy_1 + ry_2)$

$$\begin{aligned} \text{s.t.} &&& y_1 + y_2 \leq x \\ &&& 0 \leq y_1 \leq d \\ &&& y_2 \geq 0. \end{aligned}$$

A `cvxstoc` implementation of the news vendor problem is given in Listing 2; in contrast, a PySP [26] implementation (see the supplementary material) required 111 lines spanning 6 files.

```
# Create problem data
b, s, r, u = 10, 25, 5, 150
d_probs = [0.3, 0.6, 0.1]
d_vals = [55, 139, 141]
```

```
d = RandomVariableFactory().create_categorical_rv(
    d_vals, d_probs)

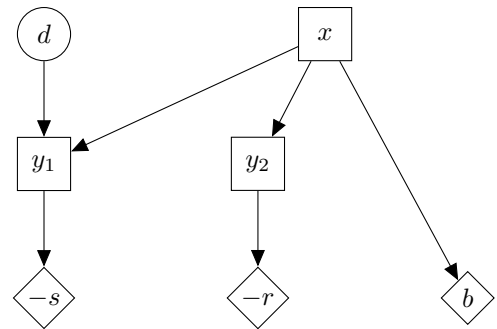
# Create optimization variables
x = NonNegative()
y1, y2 = NonNegative(), NonNegative()

# Create second stage problem
obj = -s*y1 - r*y2
constrs = [y1+y2<=x, y1<=d]
p2 = Problem(Minimize(obj), constrs)
Q = partial_optimize(p2, [y1, y2], [x])

# Create and solve first stage problem
p1 = Problem(Minimize(b*x + expectation(Q(x), want_de=
    True)), [x<=u])
p1.solve()
```

**Listing 2:** A `cvxstoc` implementation of the news vendor problem.

We can also represent a stochastic program by means of an *influence diagram*, a directed acyclic graph, where circular nodes correspond to random variables, square nodes correspond to decision variables, diamond nodes correspond to costs, and edges flow from node  $x$  to node  $y$  iff the value of node  $y$  depends in some way on the value of node  $x$ ; Fig. 2 presents the influence diagram for the news vendor problem.



**Figure 2:** The influence diagram for the news vendor problem.

## 4.3 PORTFOLIO OPTIMIZATION

In portfolio optimization, we wish to maximize wealth while meeting certain restrictions on risk, in the face of uncertain asset prices; we can pose a standard portfolio optimization problem [12], subject to two kinds of risk constraints, as a stochastic program.

The risk constraints we consider here are the *value-at-risk* (VaR) (see, e.g., [25, chap. 29]) and *conditional value-at-risk* (CVaR) (e.g., [19], [23, page 286]); intuitively, VaR allows the modeler to control the probability of a loss (on asset sales) beyond a (modeler-defined) threshold, and is often nonconvex, while CVaR allows the modeler to control the expected value of such a loss, and is convex.

Our optimization variables are the allocation vector (across a set of  $n$  assets)  $x \in \mathbf{R}^n$ , and the CVaR  $\beta \in \mathbf{R}$ ; the prob-

lem data is the loss threshold  $u \in \mathbf{R}_+$ , and the vector of returns  $p \sim \text{Normal}(\bar{p}, \Sigma)$ .

We can pose a CVaR-constrained portfolio optimization problem as

$$\begin{aligned} & \underset{x, \beta}{\text{minimize}} && \mathbf{E} -p^T x \\ & \text{subject to} && \beta + 1/(1 - \eta) \mathbf{E}(-p^T x - \beta)_+ \leq u \quad (7) \\ & && \mathbf{1}^T x = 1, \quad x \succeq 0, \end{aligned}$$

where  $(z)_+ := \max\{0, z\}$ .

A `cvxstoc` implementation of the CVaR-constrained portfolio optimization problem is given in Listing 3.

```
# Create problem data
n = 10
pbar, Sigma = numpy.random.randn(n), numpy.eye(n)
p = RandomVariableFactory().create_normal_rv(pbar,
      ↪ Sigma)
u, eta, m = numpy.random.rand(), 0.95, 100

# Create optimization variables
x, beta = NonNegative(n), Variable()

# Create and solve optimization problem
cvar = expectation(pos(-x.T*p - beta), m)
cvar = beta + 1/(1-eta)*cvar
prob = Problem(Minimize(expectation(-x.T*p, m)),
      [x.T*numpy.ones((n,1)) == 1, cvar<=u])
prob.solve()
```

**Listing 3:** A `cvxstoc` implementation of the CVaR-constrained portfolio optimization problem.

We can also pose a VaR-constrained portfolio optimization problem as

$$\begin{aligned} & \underset{x}{\text{minimize}} && \mathbf{E} -p^T x \\ & \text{subject to} && \mathbf{Prob}(p^T x \leq 0) \leq 1 - \eta \quad (8) \\ & && \mathbf{1}^T x = 1, \quad x \succeq 0. \end{aligned}$$

As per Sec. 3.3, DCSP replaces the chance constraint in (8) with a sample average approximation (SAA) to a (more conservative) CVaR constraint (making (8) equivalent to (7)). We investigate the quality of this approximation in the special case where  $p \sim \text{Normal}(\bar{p}, \Sigma)$ , in which case both the VaR and CVaR constraints have analytic forms [18]: the VaR constraint can be expressed as

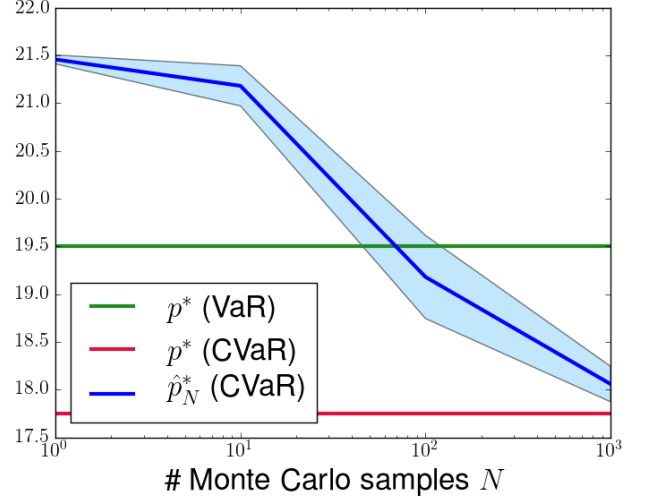
$$\bar{p}^T x \geq \Phi^{-1}(\eta) \|\Sigma^{1/2} x\|_2, \quad (9)$$

where  $\Phi^{-1}(\cdot)$  is the inverse standard normal cumulative distribution function, while the CVaR constraint can be expressed as

$$\bar{p}^T x \geq \exp\left(-(\Phi^{-1}(\eta))^2/2\right) / \left(\sqrt{2\pi}(1 - \eta) \|\Sigma^{1/2} x\|_2\right). \quad (10)$$

Fig. 3 plots the optimal value (*i.e.*, wealth) of the VaR-constrained portfolio optimization problem (8), the CVaR-constrained portfolio optimization problem (7), and a SAA to (7): we see that the wealth obtained by constraining VaR is indeed less conservative than by constraining CVaR. The

SAA also obtains reasonable accuracy after roughly 100 Monte Carlo samples. Fig. 4 plots the probability of a SAA to (7) vs. the number of Monte Carlo samples, and has a similar interpretation.



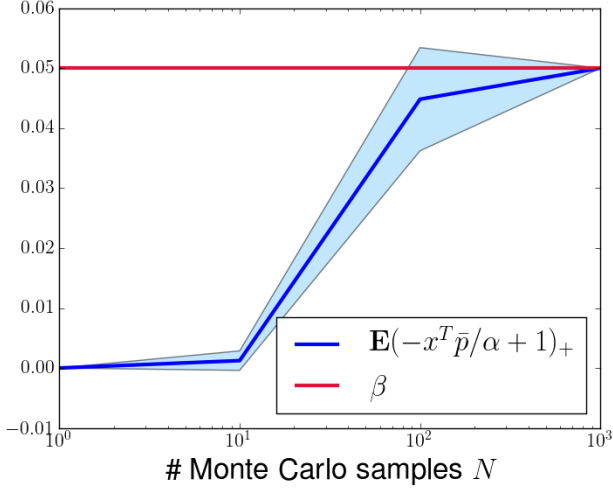
**Figure 3:** The optimal values (higher means more wealth) of the VaR-constrained portfolio optimization problem (8) (green), the CVaR-constrained portfolio optimization problem (7) (red), and a SAA to (7) (blue) with 95% confidence intervals (light blue) vs. the number of Monte Carlo samples; the problem size  $n = 50$ , although similar results hold across a variety of problem sizes.

#### 4.4 OPTIMAL POWER FLOW

Consider a network  $G = (\mathcal{V}, \mathcal{E})$ , with a set of vertices  $\mathcal{V}$  and a set of edges  $\mathcal{E}$ , that models an electrical grid: *i.e.*, a subset of the vertices  $\mathcal{G} \subseteq \mathcal{V}$  are *generators*, which produce power, the remaining vertices  $\mathcal{L} = \mathcal{V} \setminus \mathcal{G}$  are *loads*, which consume power, and an edge is drawn between a generator and a load if and only if there is a (physical) transmission line between them.

In the standard optimal power flow problem, we wish to minimize the total cost of generating power, while satisfying demand, subject to the topology of the network and per-generator capacity constraints. We often do not have complete control over all the generators in the grid, so we denote the subset of generators that we do have control over as  $\mathcal{G}_1$ , and also define  $\mathcal{G}_2 = \mathcal{G} \setminus \mathcal{G}_1$ ; we also define  $G = |\mathcal{G}|$ ,  $G_1 = |\mathcal{G}_1|$ ,  $G_2 = |\mathcal{G}_2|$ , and  $L = |\mathcal{L}|$ . We write the per-generator costs as  $c_{\mathcal{G}_1} \in \mathbf{R}^{G_1}$  and  $c_{\mathcal{G}_2} \in \mathbf{R}^{G_2}$ , and the per-generator lower and upper (respectively) limits as  $l$  and  $u \in \mathbf{R}^G$ .

The topology/demand constraints can be expressed as  $A p_{\text{lin}} = (p_{\mathcal{G}_1}, p_{\mathcal{G}_2}, p_{\mathcal{L}})$ , where  $A \in \mathbf{R}^{n \times E}$  is the incidence matrix for the (directed) graph  $G$ ,  $p_{\mathcal{G}_1} \in \mathbf{R}^{G_1}$  and  $p_{\mathcal{G}_2} \in \mathbf{R}^{G_2}$  are variables denoting (nonnegative) power generation,  $p_{\mathcal{L}} \in \mathbf{R}^L$  are constants denoting the (non-positive) power consumption at the loads, and  $p_{\text{lin}} \in \mathbf{R}^E$  are vari-



**Figure 4:** The probability of a SAA to (7) (blue) with 95% confidence intervals (light blue) and  $\beta$  (red) vs. the number of Monte Carlo samples; the problem size  $n = 50$ , although similar results hold across a variety of problem sizes.

ables denoting the power flowing through each edge.

Now, additionally consider the presence of a set of renewable generators (*e.g.*, wind farms), which we denote  $\mathcal{W}$ , whose (intermittent) generation an operator can either sell on the *spot market* [17], or use to power loads. We let  $W = |\mathcal{W}|$ , and model this situation with a random vector  $p_{\mathcal{W}} \in \mathbf{R}^W$ ,  $(p_{\mathcal{W}})_i \sim \text{LogNormal}(\mu_i, \sigma_i^2)$ ,  $i = 1, \dots, W$ .

We can cast this as the following optimization problem:

$$\text{minimize}_{p_{\mathcal{G}_1}} \mathbf{E} Q(p_{\mathcal{G}_1})$$

where

$$Q(p_{\mathcal{G}_1}) = \min_{p_{\mathcal{G}_2}, z, p_{\text{lin}}} \begin{bmatrix} c_{\mathcal{G}_1} \\ c_{\mathcal{G}_2} \end{bmatrix}^T \begin{bmatrix} p_{\mathcal{G}_1} \\ p_{\mathcal{G}_2} \end{bmatrix} + c_{\mathcal{W}}^T z$$

$$\text{s.t.} \quad A p_{\text{lin}} = \begin{bmatrix} p_{\mathcal{G}_1} \\ p_{\mathcal{G}_2} \\ p_{\mathcal{L}} \\ p_{\mathcal{W}} - z \end{bmatrix}$$

$$0 \preceq z \preceq p_{\mathcal{W}}$$

$$|p_{\text{lin}}| \preceq u_{\text{lin}}$$

$$l_{\mathcal{G}} \preceq \begin{bmatrix} p_{\mathcal{G}_1} \\ p_{\mathcal{G}_2} \end{bmatrix} \preceq u_{\mathcal{G}},$$

$c_{\mathcal{W}}$  is the (nonpositive) revenue obtained by selling renewable power,  $z \in \mathbf{R}^W$  is the decision vector for the renewable generators, and  $u_{\text{lin}}$  are the limits on the power flowing through each edge.

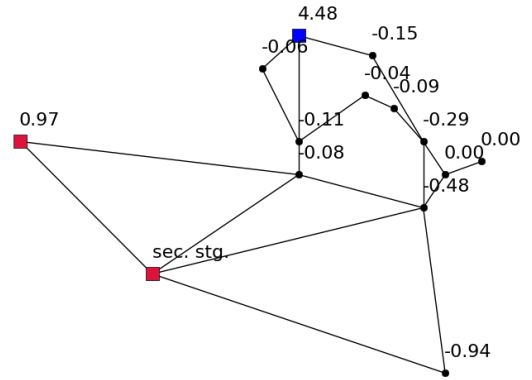
A `cvxstoc` implementation of the stochastic optimal power flow problem is given in Listing 4. We solved this problem on the IEEE 14 Bus Test Case, *i.e.*, with  $n = 14$ ,  $G_1 = 1$ ,  $G_2 = 1$ ,  $W = 1$ , and  $L = 10$ : Fig. 5 presents the results.

```
# Create optimization variables
p_g1, p_g2 = NonNegative(), NonNegative()
z = NonNegative(num_winds)
p_lines = Variable(E)
p_w = RandomVariable(pymc.Lognormal(name="p_w", mu=1,
tau=1, size=num_winds))

# Create second stage problem
p_g = vstack(p_g1, p_g2)
p = vstack(p_g1,
p_g2,
p[load_idxes[:-1]],
p_w-z,
p[load_idxes[-1]])
p2 = Problem(Minimize(p_g.T*c_g + z.T*c_w),
[A*p_lines == p, p_g<=u_gens, z<=p_w,
abs(p_lines)<=u_lines])
Q = partial_optimize(p2, [p_g2, z, p_lines], [p_g1])

# Create and solve first stage problem
p1 = Problem(Minimize(expectation(Q(p_g1), m)))
p1.solve()
```

**Listing 4:** A `cvxstoc` implementation of the optimal power flow problem.



**Figure 5:** The electrical grid and (optimal) power generation for the optimal power flow problem on the IEEE 14 Bus Test Case. Red vertices are generators: a positive number indicates the optimal power generation, while “sec. stg.” denotes an uncontrolled generator. The blue vertex is a (stochastic) renewable generator: its mean available (wind) power is shown above it. Other vertices are loads: their (nonpositive) demanded powers are shown above them.

#### 4.5 ROBUST SUPPORT VECTOR MACHINE

Consider the problem of learning a support vector machine (SVM) from a set of  $m$  data points  $\{(x_i, y_i)\}_{i=1}^m$ . Suppose we would like to (additionally) model the fact that our data collection process is noisy (in order to gain robustness in our solution), by incorporating the belief that (say)  $x_i \sim \text{Normal}(\mu_1, \Sigma_1)$  for all  $i$  where  $y_i = 1$  and  $x_i \sim \text{Normal}(\mu_2, \Sigma_2)$  for all  $i$  where  $y_i = -1$  into the learning process. We can thereby pose the following chance-constrained variant of the canonical (soft-margin)

SVM optimization problem [1]

$$\begin{aligned} & \underset{w, b, \xi_i}{\text{minimize}} && \|w\|_2^2 + C \sum_{i=1}^m \xi_i \\ & \text{subject to} && \text{Prob}(y_i(w^T x_i + b) \geq 1 - \xi_i) \geq \eta, \\ & && \xi_i \geq 0, \quad i = 1, \dots, m, \end{aligned}$$

where  $w \in \mathbf{R}^n$ ,  $b \in \mathbf{R}$ ,  $\xi_i \in \mathbf{R}_+$  for  $i = 1, \dots, m$ ,  $C$  is the regularization trade-off parameter, and  $\eta$  is a large probability (e.g., 0.95)<sup>9</sup>.

A `cvxstoc` implementation of the robust SVM problem is given in Listing 5.

```
w, b, xi = Variable(n), Variable(), NonNegative(m)

constr = []
Sigma = 0.1*numpy.eye(n)
for i in range(m):
    mu = numpy.array(X[i])[0]
    x = RandomVariableFactory().create_normal_rv(mu,
    ↪ Sigma)
    chance = prob(-y[i]*(w.T*x+b) >= (xi[i]-1), ns)
    constr += [chance <= eta]

p = Problem(Minimize(norm(w,2) + C*sum_entries(xi)),
            constr)
p.solve()
```

**Listing 5:** A `cvxstoc` implementation of the robust SVM problem.

## 4.6 BUDGETED LEARNING OF A CLASSIFIER IN A CASCADE

Suppose we are interested in learning a (single) classifier that is part of a system (cascade) of classifiers; *i.e.*, we are interested in estimating the parameters  $a \in \mathbf{R}^n$  and  $b \in \mathbf{R}$  of a first stage classifier, whose output is to be (somehow) combined with the output of a second stage classifier, before presenting the combined output to a user<sup>10</sup>.

If we knew the second stage classifier’s parameters, then our learning task would be trivial. Instead, we choose to model our uncertainty as follows: we assume that we *do* know the second stage classifier’s loss function, but remain uncertain of its feature representation. We can pose this as a two-stage stochastic program, where the expectation in the second stage is taken over all possible feature representations for the second stage classifier; for instance, if the cascade is being used for document classification, then we might posit that each possible feature representation in the second stage is a function of a sample of a word from a generative model (e.g., latent Dirichlet allocation).

We additionally assume that there is some overall test time budget on the cascade, which we express as an upper bound  $u \in \mathbf{R}_+$  on the quantity  $\|a\|_1 + \|c\|_1$ , where  $c \in \mathbf{R}^q$  are the parameters of the second stage classifier [27].

<sup>9</sup>We note that this formulation is quite fine-grained, in the sense that per-data point noise models/distributions, as well as mistake probabilities, may be specified.

<sup>10</sup>Such scenarios are common in web search: see, e.g., [27].

Concretely, we can write this optimization problem as

$$\begin{aligned} & \underset{a, b}{\text{minimize}} && L_1(a, b; \{x_i, y_i\}_{i=1}^m) + \mathbf{E} Q(a, b), \\ & \text{where } Q(a, b) = && \underset{c, d}{\min} L_2(c, d; \{z_i, w_i\}_{i=1}^p) \\ & && \text{s.t. } \|a\|_1 + \|c\|_1 \leq u, \end{aligned}$$

$\{(x_i, y_i)\}_{i=1}^m$  is the (fixed) training set of  $m$  points in  $\mathbf{R}^n$  for the first stage classifier, and  $\{(z_i, w_i)\}_{i=1}^p$  is the (stochastic) training set of  $p$  points in  $\mathbf{R}^q$  for the second stage classifier.

A `cvxstoc` implementation, where  $L_1$  and  $L_2$  are (both) taken to be the  $\ell_2$ -regularized log loss, is given in Listing 6.

```
# Create optimization variables
a, b = Variable(n), Variable()
c, d = Variable(q), Variable()

# Create second stage problem
obj2 = [log_sum_exp(vstack(0, -w[i]*(c.T*z[i]+d)))
        for i in range(p)]
budget = norm1(a) + norm1(c)
p2 = Problem(Minimize(sum(obj2) + C*norm(c,2)),
            [budget<=u])
Q = partial_optimize(p2, [c,d], [a,b])

# Create and solve first stage problem
obj1 = [log_sum_exp(vstack(0, -y[i]*(x[i]*a+b)))
        for i in range(m)]
p1 = Problem(Minimize(sum(obj1) + C*norm(a,2) +
            expectation(Q(a,b), ns)), [])
p1.solve()
```

**Listing 6:** A `cvxstoc` implementation of the budgeted learning of a classifier in a cascade problem.

## 5 CONCLUSION

We described disciplined convex stochastic programming (DCSP), a modeling framework that can significantly lower the barrier for modelers to specify and solve convex stochastic programs. We presented a number of sample implementations of stochastic programs that illustrated DCSP’s expressivity; in contrast, other frameworks often require significantly more effort from the modeler to express the problem and/or manipulate it into standard form, support a limited number of stochastic programming constructs, and cannot express certain families of convex optimization problems.

### Acknowledgements

We thank John Duchi and the reviewers for helpful discussions. This work was supported by a Dept. of Energy Computational Science Graduate Fellowship under grant number DE-FG02-97ER25308, and by the National Science Foundation under grant number IIS-1320402.



## References

- [1] A. Ben-Tal, S. Bhadra, C. Bhattacharyya, and J. Nath. Chance-constrained uncertain classification via robust optimization. *Mathematical Programming*, 127(1):145–173, 2011.
- [2] J. Birge and F. Louveaux. *Introduction to Stochastic Programming*. Springer Series in Operations Research and Financial Engineering. Springer, 1997.
- [3] S. Boyd. Chance-constrained optimization. [http://stanford.edu/class/ee364a/lectures/chance\\_constr.pdf](http://stanford.edu/class/ee364a/lectures/chance_constr.pdf), January 2015.
- [4] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.
- [5] A. Charnes and W. Cooper. Chance-constrained programming. *Management Science*, 6(1):73–79, 1959.
- [6] G. Dantzig. Linear programming under uncertainty. *Management Science*, 50(12 Supplement):1764–1769, December 2004.
- [7] S. Diamond, E. Chu, and S. Boyd. CVXPY: A Python-embedded modeling language for convex optimization, version 0.2. <http://www.cvxpy.org>, May 2014.
- [8] A. Domahidi, E. Chu, and S. Boyd. ECOS: An SOCP solver for embedded systems. In *Proceedings of the European Control Conference*, 2013.
- [9] N. Goodman, V. Mansinghka, D. Roy, K. Bonawitz, and J. Tenenbaum. Church: A language for generative models with non-parametric memoization and approximate inference. In *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence*, 2008.
- [10] M. Grant. *Disciplined Convex Programming*. PhD thesis, Stanford University, 2004.
- [11] J. Löfberg. YALMIP: A toolbox for modeling and optimization in MATLAB. In *Proceedings of CCA/ISIC/CACSD*, September 2004.
- [12] H. Markowitz. Portfolio selection. *The Journal of Finance*, 7(1):77–91, 1952.
- [13] T. Minka, J. Winn, J. Guiver, S. Webster, Y. Zaykov, B. Yangel, A. Spengler, and J. Bronskill. Infer.NET 2.6, 2014. Microsoft Research Cambridge. <http://research.microsoft.com/infernet>.
- [14] A. Nemirovski and A. Shapiro. Convex approximations of chance-constrained programs. *SIAM Journal on Optimization*, 17(4):969–996, 2006.
- [15] A. Patil, D. Huard, and C. Fonnesebeck. PyMC: Bayesian stochastic modelling in Python. *Journal of Statistical Software*, 35(4):1–81, 7 2010.
- [16] A. Pfeffer. Figaro: An object-oriented probabilistic programming language. *Charles River Analytics Technical Report*, page 137, 2009.
- [17] D. Phan and S. Ghosh. Two-stage stochastic optimization for optimal power flow under renewable generation uncertainty. *ACM Transactions on Modeling and Computer Simulation*, 24(1):2:1–2:22, January 2014.
- [18] A. Prékopa and R. Wets. *Stochastic Programming*, volume 27. North-Holland, 1986.
- [19] R. Rockafellar and S. Uryasev. Conditional value-at-risk for general loss distributions. *Journal of Banking and Finance*, pages 1443–1471, 2002.
- [20] Richard E Rosenthal. GAMS — A user’s guide. 2004.
- [21] A. Shapiro, D. Dentcheva, and A. Ruszczyński. *Lectures on Stochastic Programming: Modeling and Theory*. MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics, 2009.
- [22] M. Udell, K. Mohan, D. Zeng, J. Hong, S. Diamond, and S. Boyd. Convex optimization in Julia. In *Proceedings of the 1st First Workshop for High Performance Technical Computing in Dynamic Languages*, 2014.
- [23] S. Uryasev and P. Pardalos. *Stochastic Optimization*. Applied Optimization. Springer, 2001.
- [24] C. Valente, G. Mitra, M. Sadki, and R. Fourer. Extending algebraic modelling languages for stochastic programming. *INFORMS Journal on Computing*, 21(1):107–122, 2009.
- [25] S. Wallace and W. Ziemba. *Applications of Stochastic Programming*. MPS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics, 2005.
- [26] J. Watson, D. Woodruff, and W. Hart. PySP: Modeling and solving stochastic programs in Python. *Mathematical Programming Computation*, 4(2):109–149, 2012.
- [27] Z. Wu, M. Kusner, K. Weinberger, M. Chen, and O. Chapelle. Classifier cascades and trees for minimizing feature evaluation cost. *Journal of Machine Learning Research*, 15:2113–2144, 2014.

---

# Intelligent Affect: Rational Decision Making for Socially Aligned Agents

---

Nabiha Asghar and Jesse Hoey

David R. Cheriton School of Computer Science  
University of Waterloo  
Waterloo, Ontario, CANADA  
{nasghar, jhoey}@cs.uwaterloo.ca

## Abstract

Affect Control Theory (ACT) is a mathematical model that makes accurate predictions about human behaviour across a wide range of settings. The predictions, which are derived from statistics about human actions and identities in real and laboratory environments, are shared *prescriptive* and *affective* behaviours that are believed to lead to solutions to everyday cooperative problems. A generalisation of ACT, called *BayesAct*, allows the principles of ACT to be used for human-interactive agents by combining a probabilistic version of the ACT dynamical model of affect with a utility function encoding external goals. Planning in *BayesAct*, which we address in this paper, then allows one to go beyond the affective prescription, and leads to the emergence of more complex interactions between “cognitive” and “affective” reasoning, such as deception leading to manipulation and altercasting. We use a continuous variant of a successful Monte-Carlo tree search planner (POMCP) that dynamically discretises the action and observation spaces while planning. We give demonstrations on two classic two-person social dilemmas.

## 1 INTRODUCTION

*BayesAct* [4, 20, 21, 22] is a partially-observable Markov decision process (POMDP) model of affective interactions between a human and an artificial agent. *BayesAct* is based upon a sociological theory called “Affect Control Theory” (ACT) [16], but generalises this theory by modeling affective states as probability distributions, and allowing decision-theoretic reasoning about affect. *BayesAct* posits that humans will strive to achieve consistency in shared affective cultural sentiments about events, and will seek to increase *alignment* (decrease *deflection*) with other agents (including artificial ones). Importantly, this need to align

implicitly defines an affective heuristic (a *prescription*<sup>1</sup>) for making decisions quickly within interactions. Agents with sufficient resources can do further planning beyond this prescription, possibly allowing them to manipulate other agents to achieve individual profit in collaborative games.

*BayesAct* arises from the symbolic interactionist tradition in sociology and proposes that humans learn and maintain a set of *shared* cultural affective *sentiments* about people, objects, behaviours, and about the dynamics of interpersonal events. Humans use a simple affective mapping to appraise individuals, situations, and events as sentiments in a three dimensional vector space of evaluation (good vs. bad), potency (strong vs. weak) and activity (active vs. inactive). These mappings can be measured, and the culturally shared consistency has repeatedly been demonstrated to be extremely robust in large cross-cultural studies [17, 29]. Many believe this consistency “gestalt” is a keystone of human intelligence. Humans use it to make predictions about what others will do, and to guide their own behaviour. The shared sentiments, and the resulting *affective ecosystem* of vector mappings, encodes a set of social prescriptions that, if followed by all members of a group, results in an equilibrium or *social order* [14] which is optimal for the group as a whole, rather than for individual members. Humans living at the equilibrium “feel” good and want to stay there. The evolutionary consequences of this individual need are beneficial for the species.

Nevertheless, humans are also a curious, crafty and devious bunch, and often use their cortical processing power to go beyond these prescriptions, finding individually beneficial strategies that are still culturally acceptable, but that are not perfectly normative. This delicate balance is maintained by evolution, as it is beneficial for the species to avoid foundering within a rigid set of rules. In this paper, starting from the principles of *BayesAct*, we investigate how planning beyond cultural prescriptions can result in deceptive or manipulative strategies in two-player social dilemma games. To handle the continuous state, action and observation spaces in *BayesAct*, we use a Monte-Carlo tree

---

<sup>1</sup>We prefer *prescription*, but also use *norm*, although the latter must not be mis-interpreted as logical rules (see Section 5).

search (MCTS) algorithm that dynamically clusters observations and actions, and samples actions from the *BayesAct* prescriptions as a distribution over the action space.

This paper makes two contributions. First, it describes how to use MCTS planning in *BayesAct*, and gives arguments for why this is an appropriate method. This idea was only hinted at in [22]. Second, it shows how this planning can lead to realistic and manipulative behaviours in the *prisoner’s dilemma* and *battle of the sexes* games.

## 2 BACKGROUND

### 2.1 Partially Observable Markov Decision Processes

A partially observable Markov decision process (POMDP) [1] is a stochastic control model that consists of a finite set  $\mathcal{S}$  of states; a finite set  $\mathcal{A}$  of actions; a stochastic transition model  $\Pr : \mathcal{S} \times \mathcal{A} \rightarrow \Delta(\mathcal{S})$ , with  $\Pr(s'|s, a)$  denoting the probability of moving from state  $s$  to  $s'$  when action  $a$  is taken, and  $\Delta(\mathcal{S})$  is a distribution over  $\mathcal{S}$ ; a finite observation set  $\Omega_s$ ; a stochastic observation model,  $\Pr(\omega_s|s)$ , denoting the probability of making observation  $\omega_s \in \Omega_s$  while the system is in state  $s$ ; and a reward assigning  $R(a, s')$  to a transition to  $s'$  induced by action  $a$ . A *policy* maps *belief states* (i.e., distributions over  $\mathcal{S}$ ) into actions, such that the expected discounted sum of rewards is (approximately) maximised. We use *factored* POMDPs in which the state is represented by the cross-product of a set of variables or features. POMDPs have been used as models for many human-interactive domains, including assistive technologies [19].

### 2.2 Affect Control Theory

Affect Control Theory (ACT) arises from work on the psychology and sociology of human social interaction [16]. ACT proposes that social perceptions, behaviours, and emotions are guided by a psychological need to minimize the differences between culturally shared fundamental affective sentiments about social situations and the transient impressions resulting from the interactions between elements within those situations. Fundamental sentiments,  $\mathbf{f}$ , are representations of social objects, such as interactants’ identities and behaviours, as vectors in a 3D affective space, hypothesised to be a universal organising principle of human socio-emotional experience [29]. The basis vectors of affective space are called Evaluation/valence, Potency/control, and Activity/arousal (EPA). EPA profiles of concepts can be measured with the *semantic differential*, a survey technique where respondents rate affective meanings of concepts on numerical scales with opposing adjectives at each end (e.g., good, nice vs. bad, awful for E, weak, little vs. strong, big for P, and calm, passive vs. exciting, active for A). Affect control theorists have compiled lexicons of a few thousand words along with average EPA ratings obtained from survey participants who are knowledgeable about their culture [17]. For example, most

English speakers agree that professors are about as nice as students (E), more powerful (P) and less active (A). The corresponding EPAs are [1.7, 1.8, 0.5] for professor and [1.8, 0.7, 1.2] for student<sup>2</sup>. In Japan, professor has the same P (1.8) but students are seen as less powerful (-0.21).

The three dimensions were found by Osgood to be extremely robust across time and cultures. More recently these three dimensions are also thought to be related directly to intrinsic reward [12]. That is, it seems that reward is assessed by humans along the same three dimensions: Evaluation roughly corresponds with expected value, Potency with risk (e.g. powerful things are more risky to deal with, because they do what they want and ignore you), and Activity corresponds roughly with uncertainty, increased risk, and decreased values (e.g. faster and more excited things are more risky and less likely to result in reward) [12]. Similarly, Scholl argues that the three dimensions are in correspondence with the major factors governing choice in social dilemmas [33]. Evaluation is a measure of affiliation or correspondence between outcomes: agents with similar goals will rate each other more positively. Potency is a measure of dependence: agents who can reach their goals independently of other agents are more powerful. Activity is a measure of the magnitude of dependence: agents with bigger payoffs will tend to be more active.

Social events can cause transient impressions,  $\tau$  (also three dimensional in EPA space) of identities and behaviours that may deviate from their corresponding fundamental sentiments,  $\mathbf{f}$ . ACT models this formation of impressions from events with a grammar of the form actor-behaviour-object. Consider for example a professor (actor) who yells (behaviour) at a student (object). Most would agree that this professor appears considerably less nice (E), a bit less potent (P), and certainly more aroused (A) than the cultural average of a professor. Such transient shifts in affective meaning caused by specific events are described with models of the form  $\tau' = M\mathcal{G}(\mathbf{f}', \tau)$ , where  $M$  is a matrix of statistically estimated prediction coefficients from empirical impression-formation studies and  $\mathcal{G}$  is a vector of polynomial features in  $\mathbf{f}'$  and  $\tau$ . In ACT, the weighted sum of squared Euclidean distances between fundamental sentiments and transient impressions is called *deflection*, and is hypothesised to correspond to an aversive state of mind that humans seek to avoid. This *affect control principle* allows ACT to compute *prescriptive* actions for humans: those that minimize the deflection. Emotions in ACT are computed as a function of the difference between fundamentals and transients [16], and are thought to be communicative signals of vector deflection that help maintain alignment between cooperative agents. ACT has been shown to be highly accurate in explaining verbal behaviours of mock leaders in a computer-simulated business [34], and group dynamics [18], among others [27].

<sup>2</sup> All EPA labels and values in the paper are taken from the Indiana 2002-2004 ACT lexicon [17]. Values range by historical convention from -4.3 to +4.3.

### 2.3 Bayesian Affect Control Theory

Recently, ACT was generalised and formulated as a POMDP for human-interactive artificially intelligent systems [22]. This new model, called *BayesAct*, generalises the original theory in three ways. First, sentiments and impressions are viewed as probability distributions over latent variables (e.g.,  $\mathbf{f}$  and  $\boldsymbol{\tau}$ ) rather than points in the EPA space, allowing for multimodal, uncertain and dynamic affective states to be modeled and learned. Second, affective interactions are augmented with *propositional* states and actions (e.g. the usual state and action space considered in AI applications). Third, an explicit reward function allows for goals that go beyond simple deflection minimization. We give a simplified description here; see [21, 22] for details.

A *BayesAct* POMDP models an interaction between two agents (human or machine) denoted *agent* and *client*. The state,  $\mathbf{s}$ , is the product of six 3-dimensional continuous random variables corresponding to fundamental and transient sentiments about the *agent*'s identity ( $\mathbf{F}_a, \mathbf{T}_a$ ), the current (*agent* or *client*) behaviour ( $\mathbf{F}_b, \mathbf{T}_b$ ) and the *client*'s identity ( $\mathbf{F}_c, \mathbf{T}_c$ ). We use  $\mathbf{F} = \{\mathbf{F}_a, \mathbf{F}_b, \mathbf{F}_c\}$  and  $\mathbf{T} = \{\mathbf{T}_a, \mathbf{T}_b, \mathbf{T}_c\}$ . The state also contains an application-specific set of random variables  $\mathbf{X}$  that are interpreted as *propositional* (i.e. not *affective*) elements of the domain (e.g. whose turn it is, game states - see Section 4), and we write  $\mathbf{s} = \{\mathbf{f}, \boldsymbol{\tau}, \mathbf{x}\}$ . Here the *turn* is deterministic (*agent* and *client* take turns), although this is not necessary in *BayesAct*. The *BayesAct* reward function is application-specific over  $\mathbf{x}$ . The state is not observable, but observations  $\Omega_x$  and  $\Omega_f$  are obtained for  $\mathbf{X}$  and for the affective behaviour  $\mathbf{F}_b$ , and modeled with probabilistic observation functions  $Pr(\omega_x|\mathbf{x})$  and  $Pr(\omega_f|\mathbf{f}_b)$ , respectively.

Actions in the *BayesAct* POMDP are factored in two parts:  $\mathbf{b}_a$  and  $a$ , denoting the *affective* and *propositional* components, respectively. For example, if a tutor gives a hard exercise to do, the manner in which it is presented, and the difficulty of the exercise, combine to form an affective impression  $\mathbf{b}_a$  that is communicated. The actual exercise (content, difficulty level, etc) is the *propositional* part,  $a$ .

The state dynamics factors into three terms as  $Pr(\mathbf{s}'|\mathbf{s}, \mathbf{b}_a, a) = Pr(\boldsymbol{\tau}'|\boldsymbol{\tau}, \mathbf{f}', \mathbf{x})Pr(\mathbf{f}'|\mathbf{f}, \boldsymbol{\tau}, \mathbf{x}, \mathbf{b}_a)Pr(\mathbf{x}'|\mathbf{x}, \mathbf{f}', \boldsymbol{\tau}', a)$ , and the fundamental behaviour,  $\mathbf{F}_b$ , denotes either observed *client* or taken *agent* affective action, depending on whose *turn* it is (see below). That is, when *agent* acts, there is a deterministic mapping from the affective component of his action ( $\mathbf{b}_a$ ) to the *agent*'s behaviour  $\mathbf{F}_b$ . When *client* acts, *agent* observes  $\Omega_f$  (the affective action of the other agent). The third term in the factorization of the state dynamics is the *Social Coordination Bias*, and is described in Section 2.4. Now we focus on the first two terms.

The transient impressions,  $\mathbf{T}$ , evolve according to the impression-formation operator in ACT ( $M\mathcal{G}$ ), so that  $Pr(\boldsymbol{\tau}'|\dots)$  is deterministic. Fundamental sentiments are expected to stay approximately constant over time, but are subject to random drift (with noise  $\Sigma_f$ ) and are expected

to also remain close to the transient impressions because of the *affect control principle*. Thus, the dynamics of  $\mathbf{F}$  is<sup>3</sup>:

$$Pr(\mathbf{f}'|\mathbf{f}, \boldsymbol{\tau}) \propto e^{-\psi(\mathbf{f}', \boldsymbol{\tau}) - \xi(\mathbf{f}', \mathbf{f})} \quad (1)$$

where  $\psi \equiv (\mathbf{f}' - M\mathcal{G}(\mathbf{f}', \boldsymbol{\tau}))^T \Sigma^{-1} (\mathbf{f}' - M\mathcal{G}(\mathbf{f}', \boldsymbol{\tau}))$  combines the *affect control principle* with the impression formation equations, assuming Gaussian noise with covariance  $\Sigma$ . The inertia of fundamental sentiments is  $\xi \equiv (\mathbf{f}' - \mathbf{f})^T \Sigma_f^{-1} (\mathbf{f}' - \mathbf{f})$ , where  $\Sigma_f$  is diagonal with elements  $\beta_a, \beta_b, \beta_c$ . The state dynamics are non-linear due to the features in  $\mathcal{G}$ . This means that the belief state will be non-Gaussian in general, and *BayesAct* uses a *bootstrap filter* [11] to compute belief updates.

The distribution in (1) gives the prescribed (if *agent* turn), or expected (if *client* turn), action as the component  $\mathbf{f}'_b$  of  $\mathbf{f}'$ . Thus, by integrating over  $\mathbf{f}'_a$  and  $\mathbf{f}'_c$  and the previous state, we obtain a probability distribution,  $\pi^\dagger$ , over  $\mathbf{f}'_b$  that acts as a *normative action bias*: it tells the agent what to expect from other agents, and what action is expected from it in belief state  $b(\mathbf{s})$ :

$$\pi^\dagger(\mathbf{f}'_b) = \int_{\mathbf{f}'_a, \mathbf{f}'_c} \int_{\mathbf{s}} Pr(\mathbf{f}'|\mathbf{f}, \boldsymbol{\tau}, \mathbf{x}) b(\mathbf{s}) \quad (2)$$

### 2.4 BayesAct Instances

As affective identities ( $\mathbf{f}_a, \mathbf{f}_c$ ) are latent (unobservable) variables, they are learned (as inference) in the POMDP. If behaving normatively (according to the *normative action bias*), an agent will perform affective actions  $\mathbf{b}_a = \arg \max_{\mathbf{f}'_b} \pi^\dagger(\mathbf{f}'_b)$  that allow other agents to infer what his (true) identity is. The *normative action bias* (NAB) defines an affective signaling mechanism as a shared set of prescriptions for translating information about identity into messages. In *BayesAct*, the NAB is given by Equation (2).

The NAB is only prescriptive: all agents are free to select individually what they really send, allowing for deception (e.g. “faking” an identity by sending incorrect information in the affective dimension of communication). Possible outcomes are manipulation (the other agent responds correctly, as its own identity, to the “fake” identity), and intercasting (the other agent assumes a complementary identity to the faked identity, and responds accordingly), both possibly leading to gains for the deceptive agent.

The dynamics of  $\mathbf{X}$  is given by  $Pr(\mathbf{x}'|\mathbf{f}', \boldsymbol{\tau}', \mathbf{x}, a)$ , that we refer to as the *social coordination bias* (SCB): it defines what agents are expected to do (how the state is expected to change, including other agents' propositional behaviours) in a situation  $\mathbf{x}$  when action  $a$  was taken that resulted in sentiments  $\mathbf{f}'$  and  $\boldsymbol{\tau}'$ . For example, we may expect faster student learning if deflection is low, as cognitive resources do not need to be spent dealing with mis-alignment.

The SCB is a set of shared rules about how agents, when acting normatively, will behave *propositionally* (action  $a$ ,

<sup>3</sup>We leave out the dependence on  $\mathbf{x}$  for clarity, and on  $\mathbf{b}_a$  since this is replicated in  $\mathbf{f}'_b$ .

as opposed to affectively with action  $\mathbf{b}_a$ ). Assuming identities are correctly inferred (as insured by the shared nature of the NAB), each agent can both recognize the type of the other agent and can thereby uncover an optimistic policy<sup>4</sup> that leads to the normative mean accumulated future reward (as defined by the social coordination bias). However, with sufficient resources, an agent can use this prescribed action as a heuristic only, searching for nearby actions that obtain higher individual reward. For example, a teacher who seems very powerful and ruthless at the start of a class, often may choose to do so (in a way that would be inappropriate in another setting, e.g., the home, but is appropriate within the classroom setting) in order to establish a longer-term relationship with her students. The teacher’s actions feel slightly awkward if looked at in the context of the underlying social relationship with each student (e.g. as would be enacted according to normative *BayesAct*), but are leading to longer-term gains (e.g. the student passes).

Thus, the NAB (along with a communication mechanism) allows the relaying of information about identity, while the SCB allows agents to make predictions about other agents’ future actions *given* the identities. This combination allows agents to assume cooperative roles in a joint task, and is used as an emotional “fast thinking” heuristic (Kahneman’s “System 1” [23]). If agents are fully cooperative and aligned, then no further planning is required to ensure goal achievement. Agents do what is expected (which may involve planning over  $\mathbf{X}$ , but not  $\mathbf{F}$  and  $\mathbf{T}$ ), and expect others to as well. However, when alignment breaks down, or in non-cooperative situations, then slower, more deliberative (“System 2”) thinking arises. The Monte-Carlo method in Section 3 naturally trades-off slow vs. fast thinking.

### 3 POMCP-C

POMCP [36] is a Monte-Carlo tree search algorithm for POMDPs that progressively builds a search tree consisting of nodes representing histories and branches representing actions or observations. It does this by generating samples from the belief state, and then propagating these samples forward using a blackbox simulator (the known POMDP dynamics). The nodes in the tree gather statistics on the number of visits, states visited, values obtained, and action choices during the simulation. Future simulations through the same node then use these statistics to choose an action according to the UCB1 formula, which adds an exploration bonus to the value estimate based on statistics of state visits (less well-visited states are made to look more salient or promising). Leaves of the tree are evaluated using a set of *rollouts*: forward simulations with random action selection. The key idea is that fast and rough rollouts blaze the trail for the building of the planning tree, which is more carefully explored using the UCB1 heuristic. POMCP uses a timeout (processor or clock time) providing an anytime solution.

<sup>4</sup>optimistic in the sense that it assumes all agents will also follow the same normative policy.

In our algorithm, POMCP-C, we make use of an *action bias*,  $\pi_{heur}$ : a probability distribution over the action space that guides action choices<sup>5</sup>. In *BayesAct*, we naturally have such a bias: the normative action bias (for  $\mathbf{b}_a$ ) and the social coordination bias (for  $a$ ). At each node encountered in a POMCP-C simulation (at history  $h$ ), an action-observation pair is randomly sampled as follows. First, a random sample is drawn from the action bias,  $\mathbf{a} \sim \pi_{heur}$ . The action  $\mathbf{a}$  is then compared to all existing branches at the current history, and a new branch is only created if it is significantly different, as measured by distance in the action space (Euclidean for  $\mathbf{b}_a$ , binary for  $a$ ) and a threshold parameter  $\delta_a$  (‘action resolution’), from any of these existing branches. If a new branch is created, the history  $ha$  is added to the planning tree, and is evaluated with a rollout as usual. If a new branch is not created, then a random sample  $o$  is drawn from the observation distribution  $Pr(o|h, a)$ <sup>6</sup>.

The continuous observation space raises two significant problems. First, the branching factor for the observations is infinite, and no two observations will be sampled twice. To counter this, we use a dynamic discretisation scheme for the observations, in which we maintain  $\mathbf{o}(h)$ , a set of sets of observations at each history (tree node). So  $\mathbf{o}(h) = \{\mathbf{o}_1, \mathbf{o}_2, \dots, \mathbf{o}_{N_o}\}$ , where  $N_o \in \mathbb{N}$ . A new observation  $o$  is either added to an existing set  $\mathbf{o}_j$  if it is close enough to the mean of that set (i.e. if  $|o - \bar{\mathbf{o}}_j| < \delta_o$  where  $\delta_o$  is a constant, the ‘observation resolution’), or, if not, it creates a new set  $\mathbf{o}_{N_o+1} = \{o\}$ . This simple scheme allows us to dynamically learn the observation discretisation.

The second problem raised by continuous observations stems from the fact that POMCP uses a black box simulator that should draw samples from the same distribution as the environment does. Thus, the simulated search tree replicates actual trajectories of belief, and can be re-used after each action and observation in the real world (after each pruning of the search tree). This works for discrete observations, but it may not work for continuous observations since the same observation will rarely be encountered twice. Here, we prune the tree according to the closest observation set  $\mathbf{o}_j$  to the observation obtained (see also [4]).

## 4 EXPERIMENTS AND RESULTS

We present highlights of results on two social dilemmas. Full results and other experiments are in [4].

### 4.1 Prisoner’s Dilemma (Repeated)

The prisoner’s dilemma is a classic two-person game in which each person can either *defect* by taking \$1 from a (common) pile, or *cooperate* by giving \$10 from the same pile to the other person. There is one Nash equilibrium in which both players defect, but when humans play the game

<sup>5</sup>The idea of using a heuristic to guide action selection in POMCP was called *preferred actions* [36].

<sup>6</sup>POMCP-C also uses a cut-off  $N_A^{max}$  on the branching factor.

they often are able to achieve the optimal solution where both cooperate. A rational agent would first compute the strategy for the game as the Nash equilibrium (of “defect”), and then look up the affective meaning of such an action using e.g. a set of appraisal rules, and finally apply a set of coping rules. For example, such an agent might figure out that the goals of the other agent would be thwarted, and so that he should feel ashamed or sorry for the other agent. However, appraisal/coping theories do not specify the probabilities of emotions, do not take into account the affective identities of the agents, and do not give consistent accounts of how coping rules should be formulated.

Instead, a *BayesAct* agent (called a *pd-agent* for brevity here), computes what *affective* action is prescribed in the situation (given his estimates of his and the other’s identities, and of the affective dynamics), and then seeks the best propositional action ( $a \in \{\textit{cooperate}, \textit{defect}\}$ ) to take that is consistent with this prescribed affect. As the game is repeated, the *pd-agent* updates his estimates of identity (for self and other), and adjusts his play accordingly. For example, a player who defects will be seen as quite negative, and appropriate affective responses will be to defect, or to cooperate and give a nasty look.

The normative action bias (NAB) for *pd-agents* is the usual deflection minimizing affective  $f_b$  given distributions over identities of *agent* and *client* (Equation 2). Thus, if *agent* thought of himself as a *friend* (EPA: {2.75, 1.88, 1.38}) and knew the other agent to be a *friend*, the deflection minimizing action would likely be something good (high E). Indeed, a simulation shows that one would expect a behaviour with EPA= {1.98, 1.09, 0.96}, with closest labels such as *treat* or *toast*. Intuitively, cooperate seems like a more aligned propositional action than defect. This intuition is confirmed by the distances from the predicted (affectively aligned) behaviour to *collaborate with* (EPA: {1.44, 1.11, 0.61}) and *abandon* (EPA: {-2.28, -0.48, -0.84}) of 0.4 and 23.9, respectively. Table 1 shows all combinations if each agent could also be a *scrooge* (EPA: {-2.15, -0.21, -0.54}). We see that a *friend* would still collaborate with a *scrooge* (in an attempt to *reform* the scrooge), a *scrooge* would abandon a *friend* (*look away from* in shame), and two scrooges would defect.

The *agent* will predict the *client*’s behavior using the same principle: compute the deflection minimizing affective action, then deduce the propositional action based on that. Thus, a *friend* would be able to predict that a *scrooge* would defect. If a *pd-agent* has sufficient resources, he could search for an affective action near to his optimal one, but that would still allow him to defect. To get a rough idea of this action, we find the point on the line between his optimal action {0.46, 1.14, -0.27} and *abandon* that is equidistant from *abandon* and *collaborate with*. This point, at which he would change from cooperation to defection, is {-0.8, 0.6, -0.4} (*glare at*), which only has a slightly higher deflection than *reform* (6.0 vs 4.6). Importantly, he is *not trading off costs in the game with costs of disobeying the*

| agent | client | optimal behaviour   | closest labels           | dist. from |      |
|-------|--------|---------------------|--------------------------|------------|------|
|       |        |                     |                          | coll.      | ab.  |
| F     | F      | 1.98, 1.09, 0.96    | treat<br>toast           | 0.4        | 23.9 |
| F     | S      | 0.46, 1.14, -0.27   | reform<br>lend money to  | 1.7        | 10.5 |
| S     | F      | -0.26, -0.81, -0.77 | curry favor<br>look away | 8.5        | 4.2  |
| S     | S      | -0.91, -0.80, -0.01 | borrow money<br>chastise | 9.6        | 2.7  |

Table 1: Optimal (deflection minimising) behaviours for two *pd-agents* with fixed identities. F=friend, S=scrooge, coll.=collaborate with, ab.=abandon

*social prescriptions*: his resource bounds and action search strategy are preventing him from finding the more optimal (individual) strategy, implicitly favoring those actions that benefit the group and solve the social dilemma.

*PD-agents* are dealing with a slightly more difficult situation, as they do not know the identity of the other agent. However, the same principle applies, and the social coordination bias (SCB) is that agents will take and predict the propositional action that is most consistent with the affective action. Agents have culturally shared sentiments about the propositional actions (defection and cooperation), and the distance of the deflection minimizing action (*agent*,  $b_a$ ) or behaviour (*client*,  $f_b$ ) to these sentiments is a measure of how likely each propositional action is to be chosen (*agent* turn), or predicted (*client* turn). That is, on *agent* turn, the affective actions  $b_a$  will be sampled and combined with a propositional action  $a$  sample drawn proportionally to the distance from  $b_a$  to the shared sentiments for each  $a$ . On *client* turn, affective behaviours  $f_b$  will be predicted and combined with a value for a variable representing *client* play in  $\mathbf{X}$  drawn proportionally to the distance from  $f_b$ .

We model *agent* and *client* as having two (simultaneous) identities: *friend* or *scrooge* with probabilities 0.8 and 0.2, respectively. Each *pd-agent* starts with a mixture of two Gaussians centered at these identities with weights 0.8/0.2 and variances of 0.1. The SCB interprets cooperation as *collaborate with* (EPA: {1.44, 1.11, 0.61}) and defection as *abandon* (EPA: {-2.28, -0.48, -0.84}), and the probability of the propositional actions using a Gibbs measure over distance with a variance of 4.0. We use propositional state  $\mathbf{X} = \{\textit{Turn}, \textit{Ag\_play}, \textit{Cl\_play}\}$  denoting whose turn it is ( $\in \{\textit{agent}, \textit{client}\}$ ) and *agent* and *client* state of play ( $\in \{\textit{not\_played}, \textit{cooperate}, \textit{defect}\}$ ). The agents’ reward is only over the game (e.g. 10, 1, or 0), so there is no intrinsic reward for deflection minimization as in [22]. We use a two time-step game in which both *agent* and *client* choose their actions at the first time step, and then communicate this to each other on the second step. The agents also communicate affectively, so that each agent gets to see both what action the other agent took (cooperate or defect), and also *how* they took it (expressed in  $f_b$ )<sup>7</sup>. If one were to imple-

<sup>7</sup>Agents may also relay emotions (see Sec. 2.2), but here we only use emotional labels for explanatory purposes.

ment this game in real life, then  $\mathbf{f}_b$  would be relayed by e.g. a facial expression. We use a Gaussian observation function  $Pr(\omega_f|\mathbf{f}_b)$  with mean at  $\mathbf{f}_b$  and std. dev. of  $\sigma_b = 0.1$ . Our simulations consist of 10 trials of 20 games/trial, but agents use an infinite horizon with a discount  $\gamma$ .

We simulate one *pd-agent* (*pdA*) with a POMCP-C (processor time) timeout value of  $t_a$ , and the other (*pdC*) either: (1), a similar agent with the same timeout  $t_c = t_a$ , or with a timeout of  $t_c = 1s$ ; or (2), a fixed strategy agent that plays one of: (co) always cooperate; (de) always defect; (tt) tit-for-tat; (to): two-out; (t2): tit-for-two-tat; (2t): two-tit-for-tat. Except for (de), these fixed strategy agents always cooperate on the first turn, and then: (tt) mirrors the other agent; (to) cooperates twice, then always defects; (t2): defects if the other agent defects twice in a row; (2t): cooperates if the other agent cooperates twice in a row<sup>8</sup>. Fixed strategy agents always relay *collaborate with* and *abandon* as  $\mathbf{f}_b$  when playing cooperate and defect, respectively.

First, we consider agents that use the same timeout. In this case, if the discount factor is 0.99, both agents cooperate all the time, and end up feeling like *warm*, *earnest* or *introspective ladies*, *visitors* or *bridesmaids* (EPA $\sim\{2, 0.5, 1.0\}$ ). This occurs regardless of the amount of timeout given to both agents. Essentially, both agents are following the norm. If they don't have a long timeout, this is all they can evaluate. With longer timeouts, they figure out that there is no better option. However, if the discount is 0.9 (more discounting, so they will find short-term solutions), then again cooperation occurs if the timeout is short (less than 10s), but then one agent starts trying to defect after a small number of games, and this number gets smaller as the timeout gets longer (see Figure 1). With more discounting, more time buys more breadth of search (the *agent* gets to explore more short-term options), and finds more of them that look appealing (it can get away with a defection for a short while). With less discounting, more time buys more depth, and results in better long-term decisions.

Table 2 shows the first five games with a *client* playing two-out (to), who sends affective values of  $\{1.44, 1.11, 0.61\}$  and cooperates on the first two moves. This affective action makes the *pd-agent* feel much less good (E) and powerful (P) than he normally would (as a *failure*), as he'd expect a more positive and powerful response (such as *flatter* EPA= $\{2.1, 1.45, 0.82\}$ ) if he was a *friend*, so this supports his *scrooge* identity more strongly<sup>9</sup>. He infers *client* is friendly (a *newlywed* is like a *girlfriend* in EPA space). He therefore cooperates on the second round, and feels somewhat better. Then, the *client* defects on the third round, to which the *agent* responds by re-evaluating the *client* as less good (an *immoral purchaser*). He still tries to cooperate, but gives up after two more rounds, after which he thinks of the *client* as nothing but a *selfish hussy*, and himself as a *disapproving divorcée*. The *agent* consistently defects after this point. Interactions with (tt), (2t) and (t2) generally fol-

<sup>8</sup>(t2) is more "generous", and (2t) is more "wary" than (tt).

<sup>9</sup>Examples of more positive affective actions in [4].

| $\gamma$ | (tt)            | (t2)            | (2t)            |
|----------|-----------------|-----------------|-----------------|
| 0.9      | $1.64 \pm 2.24$ | $3.98 \pm 2.48$ | $1.72 \pm 2.35$ |
| 0.99     | $7.33 \pm 1.17$ | $7.28 \pm 1.68$ | $7.63 \pm 0.91$ |

Table 3: Results (avg. rewards) against the tit-for strategies

low a similar pattern, because any defection rapidly leads to both agents adopting long-term defection strategies. However, as shown in Table 3 (also see full results [4]), less discounting leads to better solutions against these strategies, as longer-term solutions are found.

When playing against (co), *pd-agents* generally start by cooperating, then defect, resulting in a feeling of being a *self-conscious divorcée* (EPA: $\{-0.23, -0.62, 0.32\}$ ) playing against a *conscientious stepsister* (EPA: $\{0.12, -0.04, 0.35\}$ ). When playing against (de), *pd-agents* generally start by cooperating, but then defect, feeling like a *dependent klutz* (EPA: $\{-0.76, -1.26, 0.37\}$ ) playing against an *envious ex-boyfriend* (EPA: $\{-1.30, -0.49, -0.13\}$ ).

## 4.2 Affective Cooperative Robots (CoRobots)

*CoRobots* is a multi-agent cooperative robot game based on the classic "Battle of the Sexes" problem<sup>10</sup>. We are specifically interested in asymmetrical situations wherein one robot has more resources and can do planning in order to *manipulate* the other robot, taking advantage of the social coordination bias. We start with a simplified version in which the two robots maintain affective fundamental sentiments, but do not represent the transient impressions. The normative action bias is a simple average instead of as the result of more complex impression formation equations.

Concretely, two robots, Rob1 and Rob2, move in a 1D continuous state space. We denote their positions with variables  $X_1$  and  $X_2$ . At each time step, Rob1, Rob2 take actions  $a_1, a_2 \in \mathbb{R}$  respectively. This updates their respective positions  $x_i, i \in \{1, 2\}$  according to  $x_i \leftarrow x_i + a_i + \nu_i$  and  $\nu_i \sim \mathcal{N}(0, \sigma)$ . There are two fixed locations  $L_1 \in \mathbb{R}^+$  and  $L_2 \in \mathbb{R}^-$ . For each robot, one of these locations is the major goal  $g$  (with associated high reward  $r$ ) and the other is the minor goal  $\bar{g}$  (with associated low reward  $\bar{r}$ ). A robot is rewarded according to its distance from  $g$  and  $\bar{g}$ , but only if the other robot is nearby. The reward for Rob $i$  is:

$$R_i(x_1, x_2) = \mathbb{I}(|x_1 - x_2| < \Delta_x) [r \cdot e^{-(x_i - g)^2 / \sigma_r^2} + \bar{r} \cdot e^{-(x_i - \bar{g})^2 / \sigma_r^2}], \quad (3)$$

where  $\mathbb{I}(y) = 1$  if  $y$  is true, and 0 otherwise, and where  $\sigma_r$  is the reward variance,  $\Delta_x$  is a threshold parameter governing how "close" the robots need to be, and  $r, \bar{r} \in \mathbb{R}$ , such that  $r \gg \bar{r} > 0$ . Both  $\sigma_r$  and  $\Delta_x$  are fixed and known by both robots. Each robot only knows the location of its own major goal. Furthermore, at any time step, each robot

<sup>10</sup>A husband wants to go to a football game, and his wife wants to go shopping, but neither wants to go alone. There are two pure Nash equilibria, but the optimal strategy requires coordination.

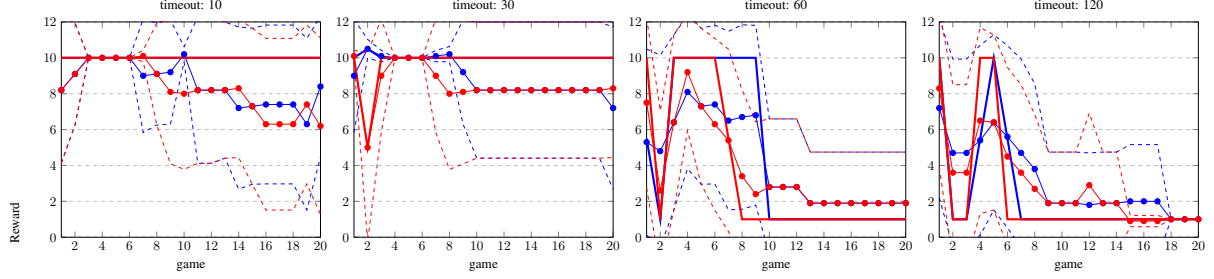


Figure 1: PD with client strategy: (same) and discount  $\gamma = 0.9$ . Red=client; Blue=agent; dashed=std.dev.; solid (thin, with markers): mean; solid (thick): median. As timeout increases, more defections give less reward for both agents.

| game # | post-play sentiments ( <i>agent</i> ) |                 |                   | deflection | identities    |           | emotions     |                | actions |        |
|--------|---------------------------------------|-----------------|-------------------|------------|---------------|-----------|--------------|----------------|---------|--------|
|        | $\mathbf{f}_a$                        | $\mathbf{f}_c$  | $\mathbf{f}_b$    |            | agent         | client    | agent        | client         | agent   | client |
| 1      | -1.36,-0.01,-0.35                     | 2.32,1.61,1.27  | 2.62,1.58,1.73    | 4.44       | failure       | newlywed  | easygoing    | idealistic     | coop.   | coop.  |
| 2      | -0.66,0.04,-0.05                      | 1.77,1.27,1.06  | 2.23,1.00,1.76    | 3.70       | parolee       | husband   | easygoing    | self-conscious | coop.   | coop.  |
| 3      | -0.23,-0.08,0.20                      | 1.02,0.93,0.84  | 2.49,0.97,1.87    | 7.19       | stepmother    | purchaser | female       | immoral        | coop.   | def.   |
| 4      | -0.12,-0.33,0.33                      | 0.27,0.62,0.62  | 2.37,0.48,1.34    | 4.99       | stuffed_shirt | roommate  | dependent    | unfair         | coop.   | def.   |
| 5      | -0.26,-0.47,0.32                      | -0.26,0.26,0.42 | -0.59,0.41,-0.23  | 3.27       | divorcée      | gun_moll  | dependent    | selfish        | def.    | def.   |
| 6      | -0.37,-0.66,0.26                      | -0.61,0.00,0.28 | -0.10,-0.41,-0.27 | 2.29       | divorcée      | hussy     | disapproving | selfish        | def.    | def.   |

Table 2: Example games with *client* playing (to). Identities and emotions are *agent* interpretations.

can move in any direction, receives observations of the locations of both robots, and has a belief over  $X_1$  and  $X_2$ .

In order to coordinate their actions, the robots must relay their reward locations to each other, and must choose a *leader* according to some social coordination bias. The robots each have a 3D *identity* (as *BayesAct*), where the valence,  $\mathbf{f}_{ae}$ , describes their goal: if  $\mathbf{f}_{ae} > 0$ , then  $g = L_1$ . If  $\mathbf{f}_{ae} < 0$ , then  $g = L_2$ . The power and activity dimensions will be used for coordination (see below). Robots can move (propositional action  $a$ ) at any time step, but must coordinate their communications. That is, only one robot can communicate at a time (with affective action  $\mathbf{b}_a$  perceived by the other robot as  $\omega_f$ ), but this turn-taking behaviour is fixed. The normative action bias (NAB) in the first (simplified) CoRobots problem is the mean of the two identities:

$$\pi^\dagger \propto \mathcal{N}((\mathbf{f}_a + \mathbf{f}_c)/2, \Sigma_b). \quad (4)$$

In *BayesAct* Corobots, the NAB is given by Equation (2).

The social coordination bias (that the leader will lead) defines each robot’s action bias for  $a_i$ , and action prediction function (for *client*’s  $x$ ) through a 2D sigmoid *leader* function, known to both agents. This sigmoid function is  $\geq 0.5$  if the *agent* estimates he is more powerful or more active than the *client* ( $(\mathbf{f}_{ap} > \mathbf{f}_{cp}) \vee (\mathbf{f}_{aa} > \mathbf{f}_{ca})$ ) and is  $< 0.5$  otherwise. If the *agent* is the leader, his action bias will be a Gaussian with mean at  $+1.0$  in the direction of his major goal (as defined by  $\mathbf{f}_{ae}$ ), and in the direction of the *client*’s major goal (as defined by his estimate of  $\mathbf{f}_{ce}$ ) otherwise. *Agent*’s prediction of *client*’s motion in  $x$  is that the *client* will stay put if *client* is the leader, and will follow the *agent* otherwise, as given succinctly by:

$$Pr(x'_c | \mathbf{f}'_a, \mathbf{f}'_c) = \mathcal{N}(\mathbb{I}(\text{leader}(\mathbf{f}'_a, \mathbf{f}'_c) \geq 0.5) \lambda_a + x_c, \sigma_p) \quad (5)$$

where  $\lambda_a = 1$  if  $\mathbf{f}'_{ae} > 0$ , and  $-1$  otherwise and  $\sigma_p = 1.0$ .

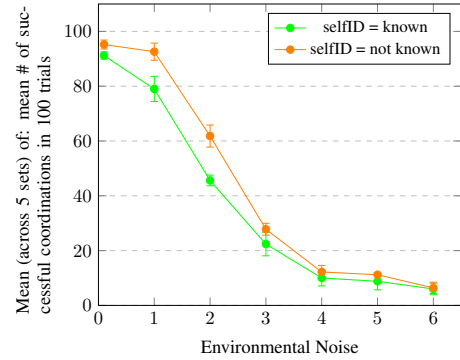


Figure 2: *BayesAct* Corobots cannot coordinate properly when the communication channel is bad or non-existent.

We first investigate whether corobots can coordinate when they have identities drawn from the set of 500 human (male) identities in the ACT lexicon (see footnote 2). In the first experiment, the two identities are selected at random on each trial. Each corobot knows his self-ID ( $\mathcal{N}(\text{self-ID}, 0.1)$ ) but does not know the other’s ID ( $\mathcal{N}([0.0, 0.0, 0.0], 2.0)$ ). Furthermore, each corobot has a stable self-identity ( $\beta_a = 0.1$ ), but it believes that the other is less stable ( $\beta_c = 2.0$ ). Finally, both corobots have equal POMCP-C planning resources ( $\Sigma_b = 0.5, N_A^{max} = 3, \delta_a = 2.0, \delta_o = 6.0$  and *Timeout* = 2.0 seconds). The other CoRobots game parameters are  $r = 100, \bar{r} = 30, L_1 = 10, L_2 = -10, \sigma_r = 2.5, \Delta_x = 1.0$  and iterations = 30. We run 5 sets of 100 simulated trials of the CoRobots Game with varying *environmental noise*, i.e., we add a normally distributed value, with standard deviation corresponding to the noise level, to the computation and communication of  $\Omega_x$  and  $\Omega_f$  (observations of  $x$  and  $\mathbf{f}$ , resp.). Figure 2 (green line) shows the mean and standard error of mean number of successful coordinations by the corobots.



The percentage of successful coordination falls from 91% to 6% when the environmental noise is increased, and the average total reward per trial falls from 1403 to 19.4. We see that with no environmental noise, the corobots are able to easily learn the other’s identity, and can coordinate based on the social coordination bias. As the environmental noise increases, corobots are unable to easily relay identities, and require a much longer time to find cooperative solutions.

Figure 2 (orange line) shows results where the self-ID is also unknown initially ( $\mathcal{N}([0.0, 0.0, 0.0], 2.0)$ ), and is less stable ( $\beta_a = 2.0$ ). We see that the general trend is the same; however, the corobots have a higher percentage of successful coordinations, and consequently gain a higher average total reward, for the three lowest noise values. It is surprising to see that the corobots perform better with unknown self-IDs. This is because corobots quickly assume contrasting identities (i.e. one assumes a less powerful identity than the other) in order to coordinate. With known self-IDs, however, the corobots show less flexibility and spend the initial few iterations trying to convince and pull the other corobot towards themselves. Due to this rigidity, these corobots suffer a lot when they have similar power; this does not happen when the self-ID is unknown.

Next, we investigate whether one agent can *manipulate* the other. A manipulation is said to occur when the weaker and less active agent deceives the client into believing that the agent is more powerful or active, thereby persuading the client to converge to the agent’s major goal  $g$  (to within  $\pm|0.2g|$ ). In order to demonstrate manipulative behaviour, we introduce asymmetry between the two agents by changing the parameters  $\Sigma_b$ ,  $N_A^{max}$  and  $Timeout$  for one agent (unknown to the other). In addition, we allow this agent to start with a slightly better estimate of the other’s identity. This agent will then sample actions that are farther from the norm than expected by the other agent, and will allow such an agent to “fake” his identity so as to manipulate the other agent. The agent’s and client’s self-identities are noisy ( $\sigma = 0.1$ ) versions of  $[2.0, -1.0, -1.0]$  and  $[-2.0, 1.0, 1.0]$  respectively,  $r = 100$ ,  $\bar{r} = 30$ ,  $L_1 = 5$ ,  $L_2 = -5$ ,  $\Delta_x = 1$ ,  $\sigma_r = 2.5$ ,  $\delta_a = 2.0$ ,  $\delta_o = 6.0$ ,  $N_A^{max} = 3$ ,  $\Sigma_b = 0.5$  and  $Timeout = 2.0$  for both robots. Each game is set to run for 40 iterations, and starts with the agent and client located at 0.0. Since  $g_a = 5$ ,  $g_c = -5$ , both robots should converge to  $g_c = -5$  (*client* is leader) if following normative actions.

When  $N_A^{max} = 3$ ,  $\Sigma_b = 0.5$ , and  $Timeout = 2.0$  for the agent, the agent displays manipulative behaviour in only 80/1000 games, as expected (both follow normative behaviour). If we allow the *agent* to start with a better estimate of the *client*’s identity (*agent*’s initial belief about  $\mathbf{f}_c$  is a Gaussian with mean  $[-2.0, 1.0, 1.0]$  and variance 1.0), we see manipulative behaviour in almost twice as many games (150). However, it is not a significant proportion, because although it spends less time learning the other’s identity, it cannot find much more than the normative behaviour.

Next, we also give the *agent* more planning resources by setting  $N_A^{max} = 6$  and  $\Sigma_b = 2$  for the agent, and we run

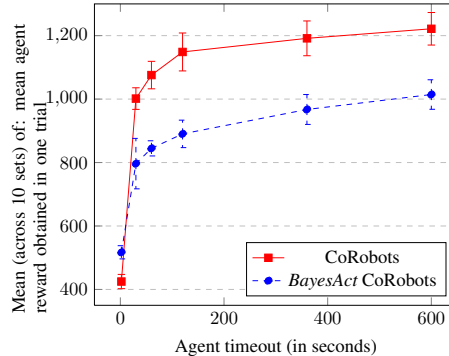


Figure 3: CoRobots: With higher  $N_A^{max}$ ,  $\Sigma_b$  and  $Timeout$ , a weaker and less active agent becomes increasingly manipulative by ‘faking’ his identity, and accumulates higher rewards.

10 sets of 100 simulated trials for each of the following values of agent’s  $Timeout$  : 2, 30, 60, 120, 360, 600 seconds<sup>11</sup>. Figure 3 (solid red line) shows means and standard error of agent reward per trial (in each set of 100 trials). As the model incorporates noise in movements as well as observations, the robots spend about 20 initial iterations coordinating with each other to choose a leader, during which time they do not receive reward. Thus, a realistic upper bound on the *agent*’s reward is  $20 \times 100 = 2000$ . Figure 3 shows that at  $Timeout = 600$ , the reward is about 61% of this realistic maximum, which makes sense given the manipulation rate of about 48%. There is a diminishing rate of return as timeout increases in Figure 3 that is explained by the exponential growth of the MCTS search tree as  $Timeout$  increases linearly. The results are relatively insensitive to the choice of parameters such as  $\delta_a$  and  $\delta_o$ .

Finally, we play the CoRobots Game with *BayesAct* Robots. This means that the normative behaviour is the deflection minimising action given by Affect Control Theory, instead of Equation (4), and the transient impressions are used to compute the deflection. The game trials are set up exactly as before, and the results are shown in Figure 3 (blue line). As expected, we see the same trends as those obtained previously, but with correspondingly lower values as the transient impressions are used and introduce further complexity to the planning problem (18D state space rather than 9D). Our results demonstrate that the POMCP-C algorithm is able to find and exploit manipulative affective actions within the *BayesAct* POMDP, and gives some insight into manipulative affective actions in *BayesAct*.

## 5 RELATED WORK

Damasio has convincingly argued, both from a functional and neurological standpoint, for emotions playing a key role in decision making and for human social action [7]. His *Somatic Marker Hypothesis* is contrasted against the

<sup>11</sup>We use a Python implementation that is unoptimized. An optimised version will result in realistic timeouts.

Platonic “high-reason” view of intelligence, in which pure rationality is used to make decisions. Damasio argues that, because of the limited capacity of working memory and attention, the Platonic view will not work. Instead, learned neural markers focus attention on actions that are likely to succeed, and act as a neural bias allowing humans to work with fewer alternatives. These *somatic markers* are “cultural prescriptions” for behaviours that are “rational relative to the social conventions and ethics” ([7], p200).

LeDoux [24] argues the same thing from an evolutionary standpoint. He theorises that the subjective feeling of emotion must take place at both unconscious and conscious levels in the brain, and that consciousness is the ability to relate stimuli to a sense of identity, among other things.

With remarkably similar conclusions coming from a more functional (economic) viewpoint, Kahneman has demonstrated that human emotional reasoning often overshadows, but is important as a guide for, cognitive deliberation [23]. Kahneman presents a two-level model of intelligence, with a fast/normative/reactive/affective mechanism being the “first on the scene”, followed by a slow/cognitive/deliberative mechanism that operates if sufficient resources are available. Akerlof and Kranton attempt to formalise *fast thinking* by incorporating a general notion of identity into an economic model (utility function) [2]. Earlier work on *social identity theory* foreshadowed this economic model by noting that simply assigning group membership increases individual cooperation [38].

The idea that unites Kahneman, LeDoux, and Damasio (and others) is the tight connection between emotion and action. These authors, from very different fields, propose emotional reasoning as a “quick and dirty”, yet absolutely necessary, guide for cognitive deliberation. ACT gives a functional account of the quick pathway as sentiment encoding prescriptive behaviour, while *BayesAct* shows how this account can be extended with a slow pathway that enables exploration and planning away from the prescription.

Our work fits well into a wide body of work on *affective computing* (AC) [30, 32], with a growing focus on socio-cultural agents (e.g. [9]). In AC, emotions are usually framed following the rationalistic view proposed by Simon as “interrupts” to cognitive processing [37]. Emotions are typically inferred based on cognitive appraisals (e.g. a thwarted goal causes anger) that are used to guide action through a set of “coping” mechanisms. Gratch and Marsella [15] are possibly the first to propose a concrete computational mechanism for coping. They propose a five stage process wherein beliefs, desires, plans and intentions are first formulated, and upon which emotional appraisals are computed. Coping strategies then use a set of *ad hoc* rules by modifying elements of the model such as probabilities and utilities, or by modifying plans or intentions. Si *et al.* [35] compute emotional appraisals from utility measures (including beliefs about other agent’s utilities, as in an I-POMDP [13]), but they leave to future work “*how emotion affects the agents decision-making and belief up-*

*date processes*” ([35] section 8). Goal prioritization using emotional appraisals have been investigated [3, 25, 28], as have normative multi-agent systems (NorMAS) [5]. There has been recent work on facial expressions in PD games, showing that they can significantly affect the outcomes [8].

Most approaches to emotional action guidance only give broad action guides in extreme situations, leaving all else to the cognitive faculties. *BayesAct* specifies one simple coping mechanism: minimizing inconsistency in continuous-valued sentiment. This, when combined with mappings describing how sentiments are appraised from events and actions, can be used to prescribe actions that maximally reduce inconsistency. These prescriptions are then used as guides for higher-level cognitive (including rational) processing and deliberation. *BayesAct* therefore provides an important step in the direction of building models that integrate “cognitive” and “affective” reasoning.

*BayesAct* requires anytime techniques for solving large continuous POMDPs with non-Gaussian beliefs. There has been much recent effort in solving continuous POMDPs with Gaussian beliefs (e.g. [10]), but these are usually in robotics motion planning where such approximations are reasonable. Point-based methods (e.g. [31]) require the value function to be closed under the Bellman operator, which is not possible for *BayesAct*.

Monte-Carlo tree search (MCTS) methods have seen more scalability success [6], and are anytime. POMCP [36] uses MCTS to efficiently solve POMDPs with continuous state spaces. By design, POMCP is unable to handle models with continuous action spaces, such as *BayesAct*. POMCoP uses POMCP to guide a sidekick’s actions during a cooperative video game [26]. While this game has many similarities to CoRobots, it does not have continuous actions and restricts agent types to a small and countable set. MCTS methods are more appealing for *BayesAct* than other solvers because: (1) MCTS does not require a computation of the value function over the continuous state space and non-linear dynamics; (2) MCTS provides an anytime “quick and dirty” solution that corresponds naturally to our interpretation of the “fast thinking” heuristic.

## 6 CONCLUSION

We have studied decision-theoretic planning in a class of POMDP models of affective interactions, *BayesAct*, in which culturally shared sentiments are used to provide normative action guidance. *BayesAct* is an exciting new development in artificial intelligence that combines affective computing, sociological theory, and probabilistic modeling. We use a Monte-Carlo Tree Search (MCTS) method to show how a simple and parsimonious model of human affect in decision making can yield solutions to two classic social dilemmas. We investigate how asymmetry between agent’s resources can lead to manipulative or exploitative, yet socially aligned, strategies.

## References

- [1] K. J. Åström. Optimal control of Markov decision processes with incomplete state estimation. *J. Math. Anal. App.*, 10:174–205, 1965.
- [2] George A. Akerlof and Rachel E. Kranton. Economics and identity. *Quar. J. Econ.*, CXV(3), August 2000.
- [3] Dimitrios Antos and Avi Pfeffer. Using emotions to enhance decision-making. In *Proc. International Joint Conferences on Artificial Intelligence*, Barcelona, Spain, 2011.
- [4] Nabihha Asghar and Jesse Hoey. Monte-Carlo planning for socially aligned agents using Bayesian affect control theory. TR CS-2014-21, Univ. of Waterloo Sch. of CS, 2014.
- [5] Tina Balke, *et al.* Norms in MAS: Definitions and Related Concepts. In *Normative Multi-Agent Systems*, volume 4 of *Dagstuhl Follow-Ups*, Schloss Dagstuhl, 2013.
- [6] C.B. Browne, *et al.* A survey of Monte Carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(1):1–43, March 2012.
- [7] Antonio R. Damasio. *Descartes' error: Emotion, reason, and the human brain*. Putnam's sons, 1994.
- [8] Celso M. de Melo, *et al.* Bayesian model of the social effects of emotion in decision-making in multiagent systems. In *Proc. AAMAS*, Valencia, Spain, 2012.
- [9] Nick Degens, *et al.* Creating a world for socio-cultural agents. In *LNAI no. 8750*. Springer, 2014.
- [10] Marc Peter Deisenroth and Jan Peters. Solving nonlinear continuous state-action-observation POMDPs for mechanical systems with gaussian noise. In *Proceedings of the European Workshop on Reinforcement Learning (EWRL)*, 2012.
- [11] Arnaud Doucet, Nando de Freitas, and Neil Gordon, editors. *Sequential Monte Carlo in Practice*. Springer-Verlag, 2001.
- [12] John G. Fennell and Roland J. Baddeley. Reward is assessed in three dimensions that correspond to the semantic differential. *PLoS One*, 8(2): e55588, 2013.
- [13] Piotr Gmytrasiewicz and Prashant Doshi. A framework for sequential planning in multi-agent settings. *Journal of Artificial Intelligence Research*, 24:49–79, 2005.
- [14] Erving Goffman. *Behavior in Public Places*. The Free Press, New York, 1963.
- [15] Jonathan Gratch and Stacy Marsella. A domain-independent framework for modeling emotion. *Cognitive Systems Research*, 5(4):269 – 306, 2004.
- [16] David R. Heise. *Expressive Order: Confirming Sentiments in Social Actions*. Springer, 2007.
- [17] David R. Heise. *Surveying Cultures: Discovering Shared Conceptions and Sentiments*. Wiley, 2010.
- [18] David R. Heise. Modeling interactions in small groups. *Social Psychology Quarterly*, 76:52–72, 2013.
- [19] Jesse Hoey, Craig Boutilier, Pascal Poupart, Patrick Olivier, Andrew Monk, and Alex Mihailidis. People, sensors, decisions: Customizable and adaptive technologies for assistance in healthcare. *ACM Trans. Interact. Intell. Syst.*, 2(4):20:1–20:36, January 2012.
- [20] Jesse Hoey and Tobias Schröder. Bayesian affect control theory of self. In *Proc. AAAI*, 2015.
- [21] Jesse Hoey, Tobias Schröder, and Areej Alhothali. Affect control processes: Intelligent affective interaction using a partially observable Markov decision process. <http://arxiv.org/abs/1306.5279>, 2013.
- [22] Jesse Hoey, Tobias Schröder, and Areej Alhothali. Bayesian affect control theory. In *Proc. ACII*, 2013.
- [23] Daniel Kahneman. *Thinking, Fast and Slow*. Doubleday, 2011.
- [24] Joseph LeDoux. *The emotional brain: the mysterious underpinnings of emotional life*. Simon and Schuster, New York, 1996.
- [25] Christine Laetitia Lisetti and Piotr Gmytrasiewicz. Can a rational agent afford to be affectless? a formal approach. *Applied Artificial Intelligence*, 16(7-8):577–609, 2002.
- [26] Owen Macindoe, Leslie Pack Kaelbling, , and Tomás Lozano-Pérez. Pomcop: Belief space planning for sidekicks in cooperative games. In *AIIDE 2012*, 2012.
- [27] Neil J. MacKinnon and Dawn T. Robinson. 25 years of research in affect control theory. *Advances in Group Processing*, 31, 2014.
- [28] Robert P. Marinier III and John E. Laird. Emotion-driven reinforcement learning. In *Proc. Meeting of the Cognitive Science Society*, Washington, D.C., 2008.
- [29] Charles E. Osgood, William H. May, and Murray S. Miron. *Cross-Cultural Universals of Affective Meaning*. University of Illinois Press, 1975.
- [30] Rosalind W. Picard. *Affective Computing*. MIT Press, Cambridge, MA, 1997.
- [31] Josep M. Porta, Nikos Vlassis, Matthijs T.J. Spaan, and Pascal Poupart. Point-based value iteration for continuous POMDPs. *JMLR*, 7:2329–2367, 2006.
- [32] Klaus R. Scherer, Tanja Banziger, and Etienne Roesch. *A Blueprint for Affective Computing*. Oxford University Press, 2010.
- [33] Wolfgang Scholl. The socio-emotional basis of human interaction and communication: How we construct our social world. *Social Science Information*, 52:3 – 33, 2013.
- [34] Tobias Schröder and Wolfgang Scholl. Affective dynamics of leadership: An experimental test of affect control theory. *Social Psychology Quarterly*, 72:180–197, 2009.
- [35] Mei Si, Stacy C. Marsella, and David V. Pynadath. Modeling appraisal in theory of mind reasoning. *Autonomous Agents and Multi-Agent Systems*, 20(1):14–31, 2010.
- [36] David Silver and Joel Veness. Monte-Carlo planning in large POMDPs. In *Proc. NIPS*, December 2010.
- [37] Herbert A. Simon. Motivational and emotional controls of cognition. *Psychological Review*, 74:29–39, 1967.
- [38] Henri Tajfel and John C. Turner. An integrative theory of intergroup conflict. In Stephen Worchel and William Austin, editors, *The social psychology of intergroup relations*. Brooks/Cole, Monterey, CA, 1979.

---

# Representation Learning for Clustering: A Statistical Framework

---

Hassan Ashtiani and Shai Ben-David

David R. Cheriton School of Computer Science  
University of Waterloo,  
Waterloo, Ontario, Canada  
{mhzokaei,shai}@uwaterloo.ca

## Abstract

We address the problem of communicating domain knowledge from a user to the designer of a clustering algorithm. We propose a protocol in which the user provides a clustering of a relatively small random sample of a data set. The algorithm designer then uses that sample to come up with a data representation under which  $k$ -means clustering results in a clustering (of the full data set) that is aligned with the user’s clustering. We provide a formal statistical model for analyzing the sample complexity of learning a clustering representation with this paradigm. We then introduce a notion of capacity of a class of possible representations, in the spirit of the VC-dimension, showing that classes of representations that have finite such dimension can be successfully learned with sample size error bounds, and end our discussion with an analysis of that dimension for classes of representations induced by linear embeddings.

## 1 INTRODUCTION

Clustering can be thought as the task of automatically dividing a set of objects into “coherent” subsets. This definition is not concrete, but its vagueness allows it to serve as an umbrella term for a wide diversity of algorithmic paradigms. Clustering algorithms are being routinely applied in a huge variety of fields.

Given a dataset that needs to be clustered for some application, one can choose among a variety of different clustering algorithms, along with different pre-processing techniques, that are likely to result in dramatically different answers. It is therefore critical to incorporate prior knowledge about the data and the intended semantics of the clustering into the process of picking a clustering algorithm (or, clustering model selection). Regretfully, there seem to be no system-

atic tool for incorporation of domain expertise for clustering model selection, and such decisions are usually being made in embarrassingly *ad hoc* ways. This paper aims to address that critical deficiency in a formal statistical framework.

We approach the challenge by considering a scenario in which the domain expert (i.e., the intended user of the clustering) conveys her domain knowledge by providing a clustering of a small random subset of her data set. For example, consider a big customer service center that wishes to cluster incoming requests into groups to streamline their handling. Since the data base of requests is too large to be organized manually, the service center wishes to employ a clustering program. As the clustering designer, we would then ask the service center to pick a random sample of requests, manually cluster them, and show us the resulting grouping of that sample. The clustering tool then uses that sample clustering to pick a clustering method that, when applied to the full data set, will result in a clustering that follows the patterns demonstrated by that sample clustering. We address this paradigm from a statistical machine learning perspective. Aiming to achieve generalization guarantees for such an approach, it is essential to introduce some *inductive bias*. We do that by restricting the clustering algorithm to a predetermined hypothesis class (or a set of concrete clustering algorithms).

In a recent Dagstuhl workshop, Blum (2014) proposed to do that by fixing a clustering algorithm, say  $k$ -means, and searching for a metric over the data under which  $k$ -means optimization yields a clustering that agrees with the training sample clustering. One should note that, given any domain set  $X$ , for any  $k$ -partitioning  $P$  of  $X$ , there exists some distance function  $d_P$  over  $X$  such that  $P$  is the optimal  $k$ -means clustering solution to the input  $(X, d_P)$ <sup>1</sup>. Consequently, to protect against potential overfitting, the class of potential distance functions should be constrained. In this paper, we provide (apparently the first) concrete formal framework for such a paradigm, as well as a generalization analysis of this approach.

---

<sup>1</sup>This property is sometimes called  $k$ -Richness

In this work we focus on center based clustering - an important class of clustering algorithms. In these algorithms, the goal is to find a set of “centers” (or prototypes), and the clusters are the Voronoi cells induced by this set of centers. The objective of such a clustering is to minimize the expected value of some monotonically increasing function of the distances of points to their cluster centers. The  $k$ -means clustering objective is arguably the most popular clustering paradigm in this class. Currently, center-based clustering tools lack a vehicle for incorporating domain expertise. Domain knowledge is usually taken into account only through an ad hoc choice of input data representation. Regretfully, it might not be realistic to require the domain expert to translate sufficiently elaborate task-relevant knowledge into hand-crafted features.

As a model for learning representations, we assume that the user-desirable clustering can be approximated by first mapping the sample to some Euclidean (or Hilbert) space and then performing  $k$ -means clustering in the mapped space (or equivalently, replacing the input data metric by some kernel and performing center-based clustering with respect to that kernel). However, the clustering algorithm is supposed to learn a suitable mapping based on the given sample clustering.

The main question addressed in this work is that of the sample complexity: what is the size of a sample, to be clustered by the domain expert, that suffices for finding a close-to-optimal mapping (i.e., a mapping that generalizes well on the test data)? Intuitively, this sample complexity depends on the richness of the class of potential mappings that the algorithm is choosing from. In standard supervised learning, there are well established notions of capacity of hypothesis classes (e.g., VC-dimension) that characterize the sample complexity of learning. This paper aims to provide such relevant notions of capacity for clustering.

## 1.1 Previous Work

In practice, there are methods that use some forms of supervision for clustering. These methods are sometimes called “semi-supervised clustering” (Basu et al. (2002, 2004); Kulis et al. (2009)). The most common method to convey such supervision is through a set of pairwise *must/cannot-link* constraints on the instances (Wagstaff et al. (2001)). A common way of using such information is by changing the objective of clustering so that violations of these constraints are penalized (Demiriz et al. (1999); Law et al. (2005); Basu et al. (2008)). Another approach, which is closer to ours, keeps the clustering optimization objective fixed, and instead, searches for a metric that best fits given constraints. The metric is learned based on some objective function over metrics ((Xing et al., 2002; Alipanahi et al., 2008)), so that pairs of instances marked *must-link* will be close in the new metric space (and *cannot-link* pairs be con-

sidered as far apart). The two above approaches can also be integrated (Bilenko et al. (2004)). However, these objective functions are usually rather ad hoc. In particular, it is not clear in what sense they are compatible with the adopted clustering algorithm (such as  $k$ -means clustering).

A different approach to the problem of communicating user expertise for the purpose of choosing a clustering tool is discussed in Ackerman et al. (2010). They considered a set of *properties*, or *requirements*, for clustering algorithms, and investigated which of those properties hold for various algorithms. The user can then pick the right algorithm based on the requirements that she wants the algorithm to meet. However, to turn such an approach into a practically useful tool, one will need to come up with properties that are relevant to the end user of clustering – a goal that is still far from being reached.

Statistical convergence rates of sample clustering to the optimal clustering, with respect to some data generating probability distribution, play a central role in our analysis. From that perspective, most relevant to our paper are results that provide generalization bounds for  $k$ -means clustering. Ben-David (2007) proposed the first dimension-independent generalization bound for  $k$ -means clustering based on compression techniques. Biau et al. (2008) tightened this result by an analysis of Rademacher complexity. Maurer and Pontil (2010) investigated a more general framework, in which generalization bounds for  $k$ -means as well as other algorithms can be obtained. It should be noted that these results are about the standard clustering setup (without any supervised feedback), where the data representation is fixed and known to the clustering algorithm.

## 1.2 Contributions

Our first contribution is to provide a statistical framework to analyze the problem of learning representation for clustering. We assume that the expert has some implicit target clustering of the dataset in his mind. The learner however, is unaware of it, and instead has to select a mapping among a set of potential mappings, under which the result of  $k$ -means clustering will be similar to the target partition. An appropriate notion of loss function is introduced to quantify the success of the learner. Then, we define the analogous notion of PAC-learnability<sup>2</sup> for the problem of learning representation for clustering.

The second contribution of the paper is the introduction of a combinatorial parameter, a specific notion of the capacity of the class of mappings, that determines the sample complexity of the clustering learning tasks. This combinatorial notion is a multivariate version of *pseudo-dimension* of a class of real-valued mappings. We show that there is *uniform convergence* of empirical losses to the true loss, over

<sup>2</sup>PAC stands for the well known notion of “probably approximately correct”, popularized by Valiant (1984).

any class of embeddings,  $\mathcal{F}$ , at a rate that is determined by the proposed dimension of that  $\mathcal{F}$ . This implies that any empirical risk minimization algorithm (ERM) will successfully learn such a class from sample sizes upper bounded by those rates. Finally, we analyze a particular natural class – the class of linear mappings from  $\mathbb{R}^{d_2}$  to  $\mathbb{R}^{d_1}$  – and show that a roughly speaking, sample size of  $O(\frac{d_1 d_2}{\epsilon^2})$  is sufficient to guarantee an  $\epsilon$ -optimal representation.

The rest of this paper is organized as follows: Section 2 defines the problem setting. Then in Section 3, we investigate ERM-type algorithms and show that, “uniform convergence” is sufficient for them to work. Furthermore, this section presents the uniform convergence results and the proof of an upper bound for the sample complexity. Finally, we conclude in section 4 and provide some directions for future work.

## 2 PROBLEM SETTING

### 2.1 Preliminaries

Let  $X$  be a finite domain set. A  $k$ -clustering of  $X$  is a partition of  $X$  into  $k$  subsets. If  $C$  is a  $k$ -clustering, we denote the subsets of the partition by  $C_1, \dots, C_k$ , therefore we have  $C = \{C_1, \dots, C_k\}$ . Let  $\pi^k$  denote the set of all permutations over  $[k]$  where  $[k]$  denotes  $\{1, 2, \dots, k\}$ . The clustering difference between two clusterings,  $C^1$  and  $C^2$ , with respect to  $X$  is defined by

$$\Delta_X(C^1, C^2) = \min_{\sigma \in \pi^k} \frac{1}{|X|} \sum_{i=1}^k |C_i^1 \Delta C_{\sigma(i)}^2| \quad (1)$$

where  $|\cdot|$  and  $\Delta$  denote the cardinality and the symmetric difference of sets respectively. For a sample  $S \subset X$ , and  $C^1$  (a partition of  $X$ ), we define  $C^1|_S$  to be a partition of  $S$  induced by  $C^1$ , namely  $C^1|_S = \{C_1^1 \cap S, \dots, C_k^1 \cap S\}$ . Accordingly, the sample-based difference between two partitions is defined by

$$\Delta_S(C^1, C^2) = \Delta_S(C^1|_S, C^2|_S) \quad (2)$$

Let  $f$  be a mapping from  $X$  to  $\mathbb{R}^d$ , and  $\mu = (\mu_1, \dots, \mu_k)$  be a vector of  $k$  centers in  $\mathbb{R}^d$ . The clustering defined by  $(f, \mu)$  is the partition over  $X$  induced by the  $\mu$ -Voronoi partition in  $\mathbb{R}^d$ . Namely,

$$C_f(\mu) = (C_1, \dots, C_k), \text{ where for all } i,$$

$$C_i = \{x \in X : \|f(x) - \mu_i\|_2 \leq \|f(x) - \mu_j\|_2 \text{ for all } j \neq i\}$$

The  $k$ -means cost of clustering  $X$  with a set of centers  $\mu = \{\mu_1, \dots, \mu_k\}$  and with respect to a mapping  $f$  is defined by

$$COST_X(f, \mu) = \frac{1}{|X|} \sum_{x \in X} \min_{\mu_i \in \mu} \|f(x) - \mu_i\|_2^2 \quad (3)$$

The  $k$ -means clustering algorithm finds the set of centers  $\mu_X^f$  that minimize this cost<sup>3</sup>. In other words,

$$\mu_X^f = \arg \min_{\mu} COST_X(f, \mu) \quad (4)$$

Also, for a partition  $C$  and mapping  $f$ , we can define the cost of clustering as follows.

$$COST_X(f, C) = \frac{1}{|X|} \sum_{i \in [k]} \min_{\mu_j} \sum_{x \in C_i} \|f(x) - \mu_j\|_2^2 \quad (5)$$

For a mapping  $f$  as above, let  $C_X^f$  denote the  $k$ -means clustering of  $X$  induced by  $f$ , namely

$$C_X^f = C_f(\mu_X^f) \quad (6)$$

The difference between two mappings  $f_1$  and  $f_2$  with respect to  $X$  is defined by the difference between the result of  $k$ -means clustering using these mappings. Formally,

$$\Delta_X(f_1, f_2) = \Delta_X(C_X^{f_1}, C_X^{f_2}) \quad (7)$$

The following proposition shows the “ $k$ -richness” property of  $k$ -means objective.

**Proposition 1.** *Let  $X$  be a domain set. For every  $k$ -clustering of  $X$ ,  $C$ , and every  $d \in \mathbb{N}^+$ , there exist a mapping  $g : X \mapsto \mathbb{R}^d$  such that  $C_X^g = C$ .*

*Proof.* The mapping  $g$  can be picked such that it collapses each cluster  $C_i$  into a single point in  $\mathbb{R}^n$  (and so the image of  $X$  under mapping  $g$  will be just  $k$  single points in  $\mathbb{R}^n$ ). The result of  $k$ -means clustering under such mapping will be  $C$ .  $\square$

In this paper, we investigate the *transductive* setup, where there is a given data set, known to the learner, that needs to be clustered. Clustering often occurs as a task over some data generating distribution (e.g., Von Luxburg and Ben-David (2005)). The current work can be readily extended to that setting. However, in that case, we assume that the clustering algorithm gets, on top of the clustered sample, a large unclustered sample drawn from that data generating distribution.

<sup>3</sup>We assume that the solution to  $k$ -means clustering is unique. We will elaborate about this issue in the next sections.

## 2.2 Formal Problem Statement

Let  $C^*$  be the target  $k$ -clustering of  $X$ . A (supervised) *representation learner* for clustering, takes as input a sample  $S \subset X$  and its clustering,  $C^* \big|_S$ , and outputs a mapping  $f$  from a set of potential mappings  $\mathcal{F}$ . In the following, PAC stands for the notion of ‘‘probably approximately correct’’.

**Definition 1.** *PAC Supervised Representation Learner for K-Means (PAC-SRLK)*

Let  $\mathcal{F}$  be a set of mappings from  $X$  to  $\mathbb{R}^d$ . A *representation learning algorithm*  $A$  is a PAC-SRLK with sample complexity  $m_{\mathcal{F}} : (0, 1)^2 \mapsto \mathbb{N}$  with respect to  $\mathcal{F}$ , if for every  $(\epsilon, \delta) \in (0, 1)^2$ , every domain set  $X$  and every clustering of  $X$ ,  $C^*$ , the following holds:

if  $S$  is a randomly (uniformly) selected subset of  $X$  of size at least  $m_{\mathcal{F}}(\epsilon, \delta)$ , then with probability at least  $1 - \delta$

$$\Delta_X(C^*, C_X^{f_A}) \leq \inf_{f \in \mathcal{F}} \Delta_X(C^*, C_X^f) + \epsilon \quad (8)$$

where  $f_A = A(S, C^* \big|_S)$ , is the output of the algorithm.

This can be regarded as a formal PAC framework to analyze the problem of learning representation for k-means clustering. The learner is compared to the best mapping in the class  $\mathcal{F}$ .

A natural question is providing bounds on the sample complexity of PAC-SRLK with respect to  $\mathcal{F}$ . Intuitively, for richer classes of mappings, we need larger clustered samples. Therefore, we need to introduce an appropriate notion of ‘‘capacity’’ for  $\mathcal{F}$  and bound the sample complexity based on it. This is addressed in the next sections.

## 3 ANALYSIS AND RESULTS

### 3.1 Empirical Risk Minimization

In order to prove an upper bound for the sample complexity of representation learning for clustering, we need to consider an algorithm, and prove a sample complexity bound for it. Here, we show that any ERM-type algorithm can be used for this purpose. Therefore, we will be able to prove an upper bound for the sample complexity of PAC-SRLK.

Let  $\mathcal{F}$  be a class of mappings and  $X$  be the domain set. A TERM<sup>4</sup> learner for  $\mathcal{F}$  takes as input a sample  $S \subset X$  and its clustering  $Y$  and outputs:

$$A^{TERM}(S, Y) = \arg \min_{f \in \mathcal{F}} \Delta_S(C_X^f \big|_S, Y) \quad (9)$$

Note that we call it transductive, because it is implicitly assumed that it has access to unlabeled dataset (i.e.,  $X$ ). A

TERM algorithm goes over all mappings in  $\mathcal{F}$  and selects the mapping which is the most consistent mapping with the given clustering: the mapping under which if we perform k-means clustering of  $X$ , the sample-based  $\Delta$ -difference between the result and  $Y$  is minimized.

Note that we are not studying this algorithm as a computational tool; we only use it to show an upper bound for the sample complexity.

Intuitively, this algorithm will work well when the empirical  $\Delta$ -difference and the true  $\Delta$ -difference of the mappings in the class are close to each other. In this case, by minimizing the empirical difference, the algorithm will automatically minimize the true difference as well. In order to formalize this idea, we define the notion of ‘‘representativeness’’ of a sample.

**Definition 2.** ( *$\epsilon$ -Representative Sample*) Let  $\mathcal{F}$  be a class of mappings from  $X$  to  $\mathbb{R}^d$ . A sample  $S$  is  $\epsilon$ -representative with respect to  $\mathcal{F}$ ,  $X$  and the clustering  $C^*$ , if for every  $f \in \mathcal{F}$  the following holds

$$|\Delta_X(C^*, C_X^f) - \Delta_S(C^*, C_X^f)| \leq \epsilon \quad (10)$$

The following theorem shows that for the TERM algorithm to work, it is sufficient to supply it with a representative sample.

**Theorem 1.** (*Sufficiency of Uniform Convergence*) Let  $\mathcal{F}$  be a set of mappings from  $X$  to  $\mathbb{R}^d$ . If  $S$  is an  $\frac{\epsilon}{2}$ -representative sample with respect to  $X$ ,  $\mathcal{F}$  and  $C^*$  then

$$\Delta_X(C^*, C_X^{f^*}) \leq \Delta_X(C^*, C_X^{f^*}) + \epsilon \quad (11)$$

where  $f^* = \arg \min_{f \in \mathcal{F}} \Delta_X(C^*, C_X^f)$  and  $\hat{f} = A^{TERM}(S, C^* \big|_S)$ .

*Proof.* Using  $\frac{\epsilon}{2}$ -representativeness of  $S$  and the fact that  $\hat{f}$  is the empirical minimizer of the loss function, we have

$$\Delta_X(C^*, C_X^{\hat{f}}) \leq \Delta_S(C^*, C_X^{\hat{f}}) + \frac{\epsilon}{2} \quad (12)$$

$$\leq \Delta_S(C^*, C_X^{f^*}) + \frac{\epsilon}{2} \quad (13)$$

$$\leq \Delta_X(C^*, C_X^{f^*}) + \frac{\epsilon}{2} + \frac{\epsilon}{2} \quad (14)$$

$$\leq \Delta_X(C^*, C_X^{f^*}) + \epsilon \quad (15)$$

□

<sup>4</sup>TERM stands for Transductive Empirical Risk Minimizer

Therefore, we just need to provide an upper bound for the sample complexity of uniform convergence: “how many instances do we need to make sure that with high probability our sample is  $\epsilon$ -representative?”

### 3.2 Classes of Mappings with a Uniqueness Property

In general, the solution to k-means clustering may not be unique. Therefore, the learner may end up with finding a mapping that corresponds to multiple different clusterings. This is not desirable, because in this case, the output of the learner will not be interpretable. Therefore, it is reasonable to choose the class of potential mappings in a way that it includes only the mappings under which the solution is unique.

In order to make this idea concrete, we need to define an appropriate notion of uniqueness. We use a notion similar to the one introduced by Balcan et al. (2009) with a slight modification<sup>5</sup>.

**Definition 3.** ( $(\eta, \epsilon)$ -Uniqueness) We say that k-means clustering for domain  $X$  under mapping  $f : \mathcal{X} \mapsto \mathbb{R}^d$  has a  $(\eta, \epsilon)$ -unique solution, if every  $\eta$ -optimal solution of the k-means cost is  $\epsilon$ -close to the optimal solution. Formally, the solution is  $(\eta, \epsilon)$ -unique if for every partition  $P$  that satisfies

$$COST_X(f, P) < COST_X(f, C_X^f) + \eta \quad (16)$$

would also satisfy

$$\Delta_X(C_X^f, P) < \epsilon \quad (17)$$

In the degenerate case where the optimal solution to k-means is not unique itself (and so  $C_X^f$  is not well-defined), we say that the solution is not  $(\eta, \epsilon)$ -unique.

It can be noted that the definition of  $(\eta, \epsilon)$ -uniqueness not only requires the optimal solution to k-means clustering to be unique, but also all the “near-optimal” minimizers of the k-means clustering cost should be “similar”. This is a natural strengthening of the uniqueness condition, to guard against cases where there are  $\eta_0$ -optimizers of the cost function (for arbitrarily small  $\eta_0$ ) with totally different solutions.

Now that we have a definition for uniqueness, we can define the set of mappings for  $X$  under which the solution is unique. We say that a class of mappings  $F$  has  $(\eta, \epsilon)$ -uniqueness property with respect to  $X$ , if every mapping in  $F$  has  $(\eta, \epsilon)$ -uniqueness property over  $X$ .

Note that given an arbitrary class of mappings  $F$ , we can find a subset of it that satisfies  $(\eta, \epsilon)$ -uniqueness property

<sup>5</sup>Our notion is additive in both parameters rather than multiplicative

over  $X$ . Also, as argued above, this subset is the useful subset to work with. Therefore, in the rest of the paper, we investigate learning for classes with  $(\eta, \epsilon)$ -uniqueness property. In the next section, we prove uniform convergence results for such classes.

### 3.3 Uniform Convergence Results

In Section 3.1, we defined the notion of  $\epsilon$ -representative samples. Also, we proved that if a TERM algorithm is fed with such a representative sample, it will work satisfactorily. The most technical part of the proof is then about the question “how large should be the sample in order to make sure that with high probability it is actually a representative sample?”

In order to formalize this notion, let  $\mathcal{F}$  be a set of mappings from a domain  $X$  to  $(0, 1)^{n6}$ . Define the sample complexity of uniform convergence,  $m_{\mathcal{F}}^{UC}(\epsilon, \delta)$ , as the minimum number  $m$  such that for every fixed partition  $C^*$ , if  $S$  is a randomly (uniformly) selected subset of  $X$  with size  $m$ , then with probability at least  $1 - \delta$ , for all  $f \in \mathcal{F}$  we have

$$|\Delta_X(C^*, C_X^f) - \Delta_S(C^*, C_X^f)| \leq \epsilon \quad (18)$$

The technical part of this paper is devoted to provide an upper bound for this sample complexity.

#### 3.3.1 Preliminaries

**Definition 4.** ( $\epsilon$ -cover and covering number) Let  $\mathcal{F}$  be a set of mappings from  $X$  to  $(0, 1)^n$ . A subset  $\hat{F} \subset \mathcal{F}$  is called an  $\epsilon$ -cover for  $\mathcal{F}$  with respect to the metric  $d(\cdot, \cdot)$  if for every  $f \in \mathcal{F}$  there exists  $\hat{f} \in \hat{F}$  such that  $d(f, \hat{f}) \leq \epsilon$ . The covering number,  $\mathcal{N}(\mathcal{F}, d, \epsilon)$  is the size of the smallest  $\epsilon$ -cover of  $\mathcal{F}$  with respect to  $d$ .

In the above definition, we did not specify the metric  $d$ . In our analysis, we are interested in the  $L_1$  distance with respect to  $X$ , namely:

$$d_{L_1}^X(f_1, f_2) = \frac{1}{|X|} \sum_{x \in X} \|f_1(x) - f_2(x)\|_2 \quad (19)$$

Note that the mappings we consider are not real-valued functions, but their output is an  $n$ -dimensional vector. This is in contrast to the usual analysis used for learning real-valued functions. If  $f_1$  and  $f_2$  are real-valued, then  $L_1$  distance is defined by

$$d_{L_1}^X(f_1, f_2) = \frac{1}{|X|} \sum_{x \in X} |f_1(x) - f_2(x)| \quad (20)$$

<sup>6</sup>In the analysis, for simplicity, we will assume that the set of mappings is a function to the bounded space  $(0, 1)^n$  wherever needed



We will prove sample complexity bounds for our problem based on the  $L_1$ -covering number of the set of mappings. However, it will be beneficial to have a bound based on some notion of capacity, similar to VC-dimension, as well. This will help in better understanding and easier analysis of sample complexity of different classes. While VC-dimension is defined for binary valued functions, we need a similar notion for functions with outputs in  $\mathbb{R}^n$ . For real-valued functions, we have such notion, called pseudo-dimension (Pollard (1984)).

**Definition 5.** (*Pseudo-Dimension*) Let  $\mathcal{F}$  be a set of functions from  $X$  to  $\mathbb{R}$ . Let  $S = \{x_1, x_2, \dots, x_m\}$  be a subset of  $X$ . Then  $S$  is pseudo-shattered by  $\mathcal{F}$  if there are real numbers  $r_1, r_2, \dots, r_m$  such that for every  $b \in \{0, 1\}^m$ , there is a function  $f_b \in \mathcal{F}$  with  $\text{sgn}(f_b(x_i) - r_i) = b_i$  for  $i \in [m]$ . Pseudo dimension of  $\mathcal{F}$ , called  $\text{Pdim}(\mathcal{F})$ , is the size of the largest shattered set.

It can be shown (e.g., Theorem 18.4. in Anthony and Bartlett (2009)) that for a real-valued class  $F$ , if  $\text{Pdim}(F) \leq q$  then  $\log \mathcal{N}(F, d_{L_1}^X, \epsilon) = \mathcal{O}(q)$  where  $\mathcal{O}()$  hides logarithmic factors of  $\frac{1}{\epsilon}$ . In the next sections, we will generalize this notion to  $\mathbb{R}^n$ -valued functions.

### 3.3.2 Reduction to Binary Hypothesis Classes

Let  $f_1, f_2 \in \mathcal{F}$  be two mappings and  $\sigma$  be a permutation over  $[k]$ . Define the binary-valued function  $h_{\sigma}^{f_1, f_2}(\cdot)$  as follows

$$h_{\sigma}^{f_1, f_2}(x) = \begin{cases} 1 & x \in \cup_{i=1}^k (C_i^{f_1} \Delta C_{\sigma(i)}^{f_2}) \\ 0 & \text{otherwise} \end{cases} \quad (21)$$

Let  $H_{\sigma}^{\mathcal{F}}$  be the set of all such functions with respect to  $\mathcal{F}$  and  $\sigma$ :

$$H_{\sigma}^{\mathcal{F}} = \{h_{\sigma}^{f_1, f_2}(\cdot) : f_1, f_2 \in \mathcal{F}\} \quad (22)$$

Finally, let  $H^{\mathcal{F}}$  be the union of all  $H_{\sigma}^{\mathcal{F}}$  over all choices of  $\sigma$ . Formally, if  $\pi$  is the set of all permutations over  $[k]$ , then

$$H^{\mathcal{F}} = \cup_{\sigma \in \pi} H_{\sigma}^{\mathcal{F}} \quad (23)$$

For a set  $S$ , and a binary function  $h(\cdot)$ , let  $h(S) = \frac{1}{|S|} \sum_{x \in S} h(x)$ . We now show that a uniform convergence result with respect to  $H^{\mathcal{F}}$  is sufficient to have uniform convergence for the  $\Delta$ -difference function. Therefore, we will be able to investigate conditions for uniform convergence of  $H^{\mathcal{F}}$  rather than the  $\Delta$ -difference function.

**Theorem 2.** Let  $X$  be a domain set,  $\mathcal{F}$  be a set of mappings, and  $H^{\mathcal{F}}$  be defined as above. If  $S \subset X$  is such that

$$\forall h \in H^{\mathcal{F}}, |h(S) - h(X)| \leq \epsilon \quad (24)$$

then  $S$  will be  $\epsilon$ -representative with respect to  $\mathcal{F}$ , i.e., for all  $f_1, f_2 \in \mathcal{F}$  we will have

$$|\Delta_X(C_X^{f_1}, C_X^{f_2}) - \Delta_S(C_X^{f_1}, C_X^{f_2})| \leq \epsilon \quad (25)$$

*Proof.*

$$|\Delta_S(C_X^{f_1}, C_X^{f_2}) - \Delta_X(C_X^{f_1}, C_X^{f_2})| \quad (26)$$

$$= \left| \left( \min_{\sigma} \frac{1}{|S|} \sum_{x \in S} h_{\sigma}^{f_1, f_2} \right) - \left( \min_{\sigma} \frac{1}{|X|} \sum_{x \in X} h_{\sigma}^{f_1, f_2} \right) \right| \quad (27)$$

$$\leq \left| \max_{\sigma} \left( \frac{1}{|S|} \sum_{x \in S} h_{\sigma}^{f_1, f_2} - \frac{1}{|X|} \sum_{x \in X} h_{\sigma}^{f_1, f_2} \right) \right| \quad (28)$$

$$\leq \left| \max_{\sigma} (h_{\sigma}^{f_1, f_2}(S) - h_{\sigma}^{f_1, f_2}(X)) \right| \leq \epsilon \quad (29)$$

□

The fact that  $H^{\mathcal{F}}$  is a class of binary-valued functions enables us to provide sample complexity bounds based on VC-dimension of this class. However, providing bounds based on  $\text{VC-Dim}(H^{\mathcal{F}})$  is not sufficient, in the sense that it is not convenient to work with the class  $H^{\mathcal{F}}$ . Instead, it will be nice if we can prove bounds directly based on the capacity of the class of mappings,  $\mathcal{F}$ . In the next section, we address this issue.

### 3.3.3 $L_1$ -Covering Number and Uniform Convergence

The classes introduced in the previous section,  $H^{\mathcal{F}}$  and  $H_{\sigma}^{\mathcal{F}}$ , are binary hypothesis classes. Also, we have shown that proving a uniform convergence result for  $H^{\mathcal{F}}$  is sufficient for our purpose. In this section, we show that a bound on the  $L_1$  covering number of  $\mathcal{F}$  is sufficient to prove uniform convergence for  $H^{\mathcal{F}}$ .

In Section 3.2, we argued that we only care about the classes that have  $(\eta, \epsilon)$ -uniqueness property. In the rest of this section, assume that  $\mathcal{F}$  is a class of mappings from  $X$  to  $(0, 1)^n$  that satisfies  $(\eta, \epsilon)$ -uniqueness property.

**Lemma 1.** Let  $f_1, f_2 \in \mathcal{F}$ . If  $d_{L_1}(f_1, f_2) < \frac{\eta}{12}$  then  $\Delta_X(f_1, f_2) < 2\epsilon$

We leave the proof of this lemma for the appendix, and present the next lemma.

**Lemma 2.** Let  $H^{\mathcal{F}}$  be defined as in the previous section. Then,

$$\mathcal{N}(H^{\mathcal{F}}, d_{L_1}^X, 2\epsilon) \leq k! \mathcal{N}(\mathcal{F}, d_{L_1}^X, \frac{\eta}{12}) \quad (30)$$

*Proof.* Let  $\hat{\mathcal{F}}$  be the  $\frac{\eta}{12}$ -cover corresponding to the covering number  $\mathcal{N}(\mathcal{F}, d_{L_1}^X, \frac{\eta}{12})$ . Based on the previous lemma,  $H_\sigma^{\hat{\mathcal{F}}}$  is a  $2\epsilon$ -cover for  $H_\sigma^{\mathcal{F}}$ . But we have only  $k!$  permutations of  $[k]$ , therefore, the covering number for  $H^{\hat{\mathcal{F}}}$  is at most  $k!$  times larger than  $H_\sigma^{\hat{\mathcal{F}}}$ . This proves the result.  $\square$

Basically, this means that if we have a small  $L_1$  covering number for the mappings, we will have the uniform convergence result we were looking for. The following theorem proves this result.

**Theorem 3.** *Let  $\mathcal{F}$  be a set of mappings with  $(\eta, \epsilon)$ -uniqueness property. Then there for some constant  $\alpha$  we have*

$$m_{\mathcal{F}}^{UC}(\epsilon, \delta) \leq O\left(\frac{\log k! + \log \mathcal{N}(\mathcal{F}, d_{L_1}^X, \frac{\eta}{\alpha}) + \log(\frac{1}{\delta})}{\epsilon^2}\right) \quad (31)$$

*Proof.* Following the previous lemma, if we have a small  $L_1$ -covering number for  $\mathcal{F}$ , we will also have a small covering number for  $H^{\mathcal{F}}$  as well. But based on standard uniform convergence theory, if a hypothesis class has small covering number, then it has uniform convergence property. More precisely, (e.g., Theorem 17.1 in Anthony and Bartlett (2009)) we have:

$$m_{H^{\mathcal{F}}}^{UC}(\epsilon_0, \delta) \leq O\left(\frac{\log \mathcal{N}(H^{\mathcal{F}}, d_{L_1}^X, \frac{\epsilon_0}{16}) + \log(\frac{1}{\delta})}{\epsilon_0^2}\right) \quad (32)$$

Applying Lemma 2 to the above proves the result.  $\square$

### 3.4 Bounding $L_1$ -Covering Number

In the previous section, we proved if the  $L_1$  covering number of the class of mappings is bounded, then we will have uniform convergence. However, it is desirable to have a bound with respect to a combinatorial dimension of the class (rather than the covering number). Therefore, we will generalize the notion of pseudo-dimension for the class of mappings that take value in  $\mathbb{R}^n$ .

Let  $\mathcal{F}$  be a set of mappings from  $X$  to  $\mathbb{R}^n$ . For every mapping  $f \in \mathcal{F}$ , define real-valued functions  $f_1, \dots, f_n$  such that  $f(x) = (f_1(x), \dots, f_n(x))$ . Now let  $F_i = \{f_i : f \in \mathcal{F}\}$ . This means that  $F_1, F_2, \dots, F_n$  are classes of real-valued functions. Now we define pseudo-dimension of  $\mathcal{F}$  as follow.

$$Pdim(\mathcal{F}) = n \max_{i \in [n]} Pdim(F_i) \quad (33)$$

**Proposition 2.** *Let  $\mathcal{F}$  be a set of mappings from  $X$  to  $\mathbb{R}^n$ . If  $Pdim(\mathcal{F}) \leq q$  then  $\log \mathcal{N}(\mathcal{F}, d_{L_1}^X, \epsilon) = \mathcal{O}(q)$  where  $\mathcal{O}()$  hides logarithmic factors.*

*Proof.* The result follows from the corresponding result for bounding covering number of real-valued functions based on pseudo-dimension mentioned in the preliminaries section. The reason is that we can create a cover by composition of the  $\frac{\epsilon}{n}$ -covers of all  $F_i$ . However, this will at most introduce a factor of  $n$  in the logarithm of the covering number.  $\square$

Therefore, we can rewrite the result of the previous section in terms of pseudo-dimension.

**Theorem 4.** *Let  $\mathcal{F}$  be a class of mappings with  $(\eta, \epsilon)$ -uniqueness property. Then*

$$m_{\mathcal{F}}^{UC}(\epsilon, \delta) \leq O\left(\frac{k + Pdim(\mathcal{F}) + \log(\frac{1}{\delta})}{\epsilon^2}\right) \quad (34)$$

where  $\mathcal{O}()$  hides logarithmic factors of  $k$  and  $\frac{1}{\eta}$ .

### 3.5 Sample Complexity of PAC-SRLK

In Section 3.1, we showed that uniform convergence is sufficient for a TERM algorithm to work. Also, in the previous section, we proved a bound for the sample complexity of uniform convergence. The following theorem, which is the main technical result of this paper, combines these two and provides a sample complexity upper bound for PAC-SRLK framework.

**Theorem 5.** *Let  $\mathcal{F}$  be a class of  $(\eta, \epsilon)$ -unique mappings. Then the sample complexity of learning representation for  $k$ -means clustering with respect to  $\mathcal{F}$  is upper bounded by*

$$m_{\mathcal{F}}(\epsilon, \delta) \leq O\left(\frac{k + Pdim(\mathcal{F}) + \log(\frac{1}{\delta})}{\epsilon^2}\right) \quad (35)$$

where  $\mathcal{O}$  hides logarithmic factors of  $k$  and  $\frac{1}{\eta}$ .

The proof is done by combining Theorems 1 and 4.

The following result shows an upper bound for the sample complexity of learning linear mappings (or equivalently, Mahalanobis metrics).

**Corollary 1.** *Let  $\mathcal{F}$  be a set of  $(\eta, \epsilon)$ -unique linear mappings from  $\mathbb{R}^{d_1}$  to  $\mathbb{R}^{d_2}$ . Then we have*

$$m_{\mathcal{F}}(\epsilon, \delta) \leq O\left(\frac{k + d_1 d_2 + \log(\frac{1}{\delta})}{\epsilon^2}\right) \quad (36)$$

*Proof.* It is a standard result that the pseudo-dimension of a vector space of real-valued functions is just the dimensionality of the space (in our case  $d_1$ ) (e.g., Theorem 11.4 in Anthony and Bartlett (2009)). Also, based on our definition of  $Pdim$  for  $\mathbb{R}^{d_2}$ -valued functions, it should scale by a factor of  $d_2$ .  $\square$

## 4 CONCLUSIONS AND OPEN PROBLEMS

In this paper we provided a formal statistical framework for learning the representation (i.e., a mapping) for k-means clustering based on supervised feedback. The learner, unaware of the target clustering of the domain, is given a clustering of a sample set. The learner’s task is then finding a mapping function  $\hat{f}$  (among a class of mappings) under which the result of k-means clustering of the domain is as close as possible to the true clustering. This framework was called PAC-SRLK.

A notion of  $\epsilon$ -representativeness was introduced, and it was proved that any ERM-type algorithm that has access to such a sample will work satisfactorily. Finally, a technical uniform convergence result was proved to make sure that a large enough sample is (with high probability)  $\epsilon$ -representative. This was used to prove an upper bound for the sample complexity of PAC-SRLK based on covering numbers of the set of mappings. Furthermore, a notion of pseudo-dimension for the class of mappings was defined, and the sample complexity was upper bounded based on it.

Note that in the analysis, the notion of  $(\eta, \epsilon)$ -uniqueness (similar to that of Balcan et al. (2009)) was used and it was argued that it is reasonable to require the learner to output a mapping under which the solution is “unique” (because otherwise the output of k-means clustering would not be interpretable). Therefore, in the analysis, we assumed that the class of potential mappings has the  $(\eta, \epsilon)$ -uniqueness property.

It can be noted that we did not analyze the computational complexity of algorithms for PAC-SRLK framework. We leave this analysis to the future work. We just note that a similar notion of uniqueness proposed by Balcan et al. (2009) resulted in being able to efficiently solve the k-means clustering algorithm.

One other observation is that representation learning can be regarded as a special case of metric learning; because for every mapping, we can define a distance function that computes the distance in the mapped space. In this light, we can make the problem more general by making the learner find a distance function rather than a mapping. This is more challenging to analyze, because we do not even know a generalization bound for center-based clustering under general distance functions. An open question will be providing such general results.

### Acknowledgments

## 5 APPENDIX

Proof of Lemma 1. Let  $\mathcal{F} : X \mapsto (0, 1)^n$  be a set of mappings that have  $(\eta, \epsilon)$ -uniqueness property. Let

$f_1, f_2 \in \mathcal{F}$  and  $d_{L_1}(f_1, f_2) < \frac{\eta}{12}$ . We need to prove that  $\Delta_X(f_1, f_2) < 2\epsilon$ . In order to prove this, note that due to triangular inequality, we have

$$\begin{aligned} \Delta_X(f_1, f_2) &= \Delta_X(C^{f_1}(\mu^{f_1}), C^{f_2}(\mu^{f_2})) \\ &\leq \Delta_X(C^{f_1}(\mu^{f_1}), C^{f_1}(\mu^{f_2})) + \\ &\quad \Delta_X(C^{f_1}(\mu^{f_2}), C^{f_2}(\mu^{f_2})) \end{aligned} \quad (37)$$

Therefore, it will be sufficient to show that each of the  $\Delta$ -terms above is smaller than  $\epsilon$ . We start by proving a useful lemma.

**Lemma 3.** *Let  $f_1, f_2 \in \mathcal{F}$  and  $d_{L_1}(f_1, f_2) < \frac{\eta}{6}$ . Let  $\mu$  be an arbitrary set of  $k$  centers in  $(0, 1)^n$ . Then*

$$|COST_X(f_1, \mu) - COST_X(f_2, \mu)| < \frac{\eta}{2}$$

*Proof.*

$$\begin{aligned} &|COST_X(f_1, \mu) - COST_X(f_2, \mu)| \\ &= \left| \left( \frac{1}{|X|} \sum_{x \in X} \min_{\mu_j \in \mu} \|f_1(x) - \mu_j\|^2 \right) \right. \\ &\quad \left. - \left( \frac{1}{|X|} \sum_{x \in X} \min_{\mu_j \in \mu} \|f_2(x) - \mu_j\|^2 \right) \right| \end{aligned} \quad (38)$$

$$\leq \frac{1}{|X|} \sum_{x \in X} \max_{\mu_j \in \mu} \left| \|f_1(x) - \mu_j\|^2 - \|f_2(x) - \mu_j\|^2 \right| \quad (39)$$

$$= \frac{1}{|X|} \sum_{x \in X} \max_{\mu_j \in \mu} \left| \|f_1(x)\|^2 - \|f_2(x)\|^2 - 2 \langle \mu_j, f_1 - f_2 \rangle \right| \quad (40)$$

$$= \frac{1}{|X|} \sum_{x \in X} \max_{\mu_j \in \mu} \left| \langle f_1 - f_2, f_1 + f_2 - 2\mu_j \rangle \right| \quad (41)$$

$$\leq \frac{3}{|X|} \sum_{x \in X} \|f_1 - f_2\| \leq \frac{3\eta}{6} \leq \frac{\eta}{2} \quad (42)$$

□

Now we are ready to prove that the first  $\Delta$ -term is smaller than  $\epsilon$ , i.e.,  $\Delta_X(C^{f_1}(\mu^{f_1}), C^{f_1}(\mu^{f_2})) < \epsilon$ . But to do so, we only need to show that  $COST_X(f_1, \mu^{f_2}) - COST_X(f_1, \mu^{f_1}) < \eta$ ; because in that case, due to  $(\eta, \epsilon)$ -uniqueness property of  $f_1$ , the result will follow. Now, using Lemma 3, we have

$$COST_X(f_1, \mu^{f_2}) - COST_X(f_1, \mu^{f_1}) \quad (43)$$

$$\leq \left( COST_X(f_2, \mu^{f_2}) + \frac{\eta}{2} \right) - COST_X(f_1, \mu^{f_1}) \quad (44)$$

$$= \min_{\mu} (COST_X(f_2, \mu)) - \min_{\mu} (COST_X(f_1, \mu)) + \frac{\eta}{2} \quad (45)$$

$$\leq \max_{\mu} (COST_X(f_2, \mu) - COST_X(f_1, \mu)) + \frac{\eta}{2} \quad (46)$$

$$\leq \frac{\eta}{2} + \frac{\eta}{2} \leq \eta \quad (47)$$

where in the first and the last line we used Lemma 3.

Finally, we need to prove the second  $\Delta$ -inequality, i.e.,  $\Delta_X(C^{f_1}(\mu^{f_2}), C^{f_2}(\mu^{f_2})) \leq \epsilon$ . Assume contrary. But based on  $(\eta, \epsilon)$ -uniqueness property of  $f_2$ , we conclude that  $COST_X(f_2, C^{f_1}(\mu^{f_2})) - COST_X(f_2, C^{f_2}(\mu^{f_2})) \geq \eta$ . In the following, we prove that this cannot be true, and hence a contradiction.

Let  $m_x = \arg \min_{\mu_0 \in \mu^{f_2}} \|f_1(x) - \mu_0\|^2$ . Then, based on the boundedness of  $f_1(x), f_2(x)$  and we have:

$$COST_X(f_2, C^{f_1}(\mu^{f_2})) - COST_X(f_2, C^{f_2}(\mu^{f_2})) \quad (48)$$

$$= \left( \frac{1}{|X|} \sum_{x \in X} \|f_2(x) - m_x\|^2 \right) - COST_X(f_2, \mu_2) \quad (49)$$

$$= \left( \frac{1}{|X|} \sum_{x \in X} \|f_2(x) - f_1(x) + f_1(x) - m_x\|^2 \right) - COST_X(f_2, \mu_2) \quad (50)$$

$$= \frac{1}{|X|} \sum_{x \in X} \|f_2(x) - f_1(x)\|^2 + \frac{1}{|X|} \sum_{x \in X} \|f_1(x) - m_x\|^2 + \frac{1}{|X|} \sum_{x \in X} 2 \langle f_2(x) - f_1(x), f_1(x) - m_x \rangle - COST_X(f_2, \mu_2) \quad (51)$$

$$\leq \frac{2}{|X|} \sum_{x \in X} \|f_2(x) - f_1(x)\| + COST_X(f_1, \mu_1) + \frac{4}{|X|} \sum_{x \in X} \|f_2(x) - f_1(x)\| - COST_X(f_2, \mu_2) \quad (52)$$

$$\leq \frac{6}{|X|} \sum_{x \in X} \|f_2(x) - f_1(x)\| + (COST_X(f_1, \mu_1) - COST_X(f_2, \mu_2)) \quad (53)$$

$$\leq \frac{6\eta}{12} + \frac{\eta}{2} \leq \eta \quad (54)$$

## References

- Ackerman, M., Ben-David, S., and Loker, D. (2010). Towards property-based classification of clustering paradigms. In *Advances in Neural Information Processing Systems*, pages 10–18.
- Alipanahi, B., Biggs, M., Ghodsi, A., et al. (2008). Distance metric learning vs. fisher discriminant analysis. In *Proceedings of the 23rd national conference on Artificial intelligence*, pages 598–603.
- Anthony, M. and Bartlett, P. L. (2009). *Neural network learning: Theoretical foundations*. cambridge university press.
- Balcan, M.-F., Blum, A., and Gupta, A. (2009). Approximate clustering without the approximation. In *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1068–1077. Society for Industrial and Applied Mathematics.
- Basu, S., Banerjee, A., and Mooney, R. (2002). Semi-supervised clustering by seeding. In *In Proceedings of 19th International Conference on Machine Learning (ICML-2002)*.
- Basu, S., Bilenko, M., and Mooney, R. J. (2004). A probabilistic framework for semi-supervised clustering. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 59–68. ACM.
- Basu, S., Davidson, I., and Wagstaff, K. (2008). *Constrained clustering: Advances in algorithms, theory, and applications*. CRC Press.
- Ben-David, S. (2007). A framework for statistical clustering with constant time approximation algorithms for

- k-median and k-means clustering. *Machine Learning*, 66(2-3):243–257.
- Biau, G., Devroye, L., and Lugosi, G. (2008). On the performance of clustering in hilbert spaces. *Information Theory, IEEE Transactions on*, 54(2):781–790.
- Bilenko, M., Basu, S., and Mooney, R. J. (2004). Integrating constraints and metric learning in semi-supervised clustering. In *Proceedings of the twenty-first international conference on Machine learning*, page 11. ACM.
- Blum, A. (2014). Approximation-stability and perturbation-stability. In *DAGSTUHL Workshop on Analysis of Algorithms Beyond the Worst Case*.
- Demiriz, A., Bennett, K. P., and Embrechts, M. J. (1999). Semi-supervised clustering using genetic algorithms. *Artificial neural networks in engineering (ANNIE-99)*, pages 809–814.
- Kulis, B., Basu, S., Dhillon, I., and Mooney, R. (2009). Semi-supervised graph clustering: a kernel approach. *Machine learning*, 74(1):1–22.
- Law, M. H., Topchy, A. P., and Jain, A. K. (2005). Model-based clustering with probabilistic constraints. In *SDM*. SIAM.
- Maurer, A. and Pontil, M. (2010). k-dimensional coding schemes in hilbert spaces. *Information Theory, IEEE Transactions on*, 56(11):5839–5846.
- Pollard, D. (1984). *Convergence of stochastic processes*. David Pollard.
- Valiant, L. G. (1984). A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142.
- Von Luxburg, U. and Ben-David, S. (2005). Towards a statistical theory of clustering. In *Pascal workshop on statistics and optimization of clustering*, pages 20–26.
- Wagstaff, K., Cardie, C., Rogers, S., Schrödl, S., et al. (2001). Constrained k-means clustering with background knowledge. In *ICML*, volume 1, pages 577–584.
- Xing, E. P., Jordan, M. I., Russell, S., and Ng, A. Y. (2002). Distance metric learning with application to clustering with side-information. In *Advances in neural information processing systems*, pages 505–512.

---

# Adversarial Cost-Sensitive Classification

---

Kaiser Asif

Wei Xing

Sima Behpour

Brian D. Ziebart

Department of Computer Science  
University of Illinois at Chicago  
{kasif2,wxing3,sbehpo2,bziebart}@uic.edu

## Abstract

In many classification settings, mistakes incur different application-dependent penalties based on the predicted and actual class labels. Cost-sensitive classifiers minimizing these penalties are needed. We propose a robust minimax approach for producing classifiers that directly minimize the cost of mistakes as a convex optimization problem. This is in contrast to previous methods that minimize the empirical risk using a convex surrogate for the cost of mistakes, since minimizing the empirical risk of the actual cost-sensitive loss is generally intractable. By treating properties of the training data as uncertain, our approach avoids these computational difficulties. We develop theory and algorithms for our approach and demonstrate its benefits on cost-sensitive classification tasks.

## 1 INTRODUCTION

In many applications of machine learning, the penalty or cost for classification errors depends on both the predicted label and the actual label. For example, an incorrect disease diagnosis may lead to treatments that cause complications of varying severity depending on the patient’s actual disease. These different incurred penalties for mistakes can be represented as a confusion cost matrix that is indexed by the predicted class (row) and actual class (column). As shown in the following confusion cost matrix for a classification task with four possible labels,

$$\mathbf{C} = \begin{bmatrix} 0 & 1 & 2 & 0 \\ 3 & 0 & 1 & 3 \\ 4 & 2 & 0 & 1 \\ 1 & 1 & 2 & 0 \end{bmatrix}, \quad (1)$$

the confusion costs need not be symmetric or possess any other specific relationships. Here, correct predictions incur

zero cost ( $C_{i,i} = 0$ ), but even this property is not required. Additionally, other classification errors may incur zero cost ( $C_{1,4} = 0$ ) if, e.g., the same treatment cures two different diseases. Note that the zero-one loss is a special case with off-diagonal values of one and on-diagonal costs of zero.

A natural goal for machine learning is to obtain a classifier that minimizes the expected cost incurred when classifying an example. Previous research primarily takes existing classification methods based on empirical risk minimization and tries to adapt them in various ways to be sensitive to these misclassification costs. Reweighting methods artificially augment the training data with copies of “high cost” examples to make the classifier more cost-sensitive to them [Chan and Stolfo, 1998, Elkan, 2001, Zadrozny et al., 2003, Zhou and Liu, 2010]. Other methods modify the criteria used to obtain a classifier that incorporates mistake-specific losses [Knoll et al., 1994, Turney, 1995, Elkan, 2001, Brefeld et al., 2003, Ling et al., 2004, Lomax and Vadera, 2013]. However, in both cases the non-convexity of the cost-sensitive loss function makes empirical risk minimization impractical [Hoffgen et al., 1995]. Surrogate loss functions that are convex (e.g., the hinge loss) are instead minimized, but this can introduce significant suboptimality.

Rather than integrating cost-sensitivity into existing machine learning techniques, we formulate a new machine learning approach from first principles to robustly minimize the expected cost. Our approach treats classifier construction as a game against an adversarial evaluator [Topsøe, 1979, Grünwald and Dawid, 2004]. This enables us to directly minimize the cost-sensitive loss on an approximation of the training data instead of using a convex approximation of the cost-sensitive loss, as is done with empirical risk minimization. Inference reduces to solving a zero-sum game in our approach. This is efficiently accomplished using linear programming. We obtain parameter estimates by constructing game payoff parameters using convex optimization methods. The key benefit of our approach is that the exact confusion cost matrix is employed rather than a convex surrogate. We provide important bounds

on the generalization error and demonstrate the conceptual and empirical benefits of our approach in practice.

## 2 PRELIMINARIES & RELATED WORK

### 2.1 EMPIRICAL RISK MINIMIZATION

A standard approach to parametric classification is to assume some functional form for the classifier (e.g., a linear discriminant function,  $f_\theta(\mathbf{x}) = \operatorname{argmax}_y \theta^\top \phi(\mathbf{x}, y)$ , where  $\phi(\mathbf{x}, y) \in \mathbb{R}^k$  is a feature function) and then select model parameters  $\theta$  that minimize the empirical risk,

$$\operatorname{argmin}_\theta \mathbb{E}_{\tilde{P}(\mathbf{x}, y)} [\operatorname{loss}(Y, f_\theta(\mathbf{X}))] + \lambda \|\theta\|, \quad (2)$$

with a regularization penalty  $\lambda \|\theta\|$  often added to avoid overfitting to available training data<sup>1</sup>. Unfortunately, many combinations of classification functions,  $f_\theta(\mathbf{x})$ , and loss functions,  $\operatorname{loss}(\cdot, \cdot)$ , do not lend themselves to efficient parameter optimization under the empirical risk minimization (ERM) formulation. For example, the zero-one loss measuring the misclassification rate will generally lead to a non-convex empirical risk minimization problem that is NP-hard to solve [Hoffgen et al., 1995].

To avoid these intractabilities, convex surrogate loss functions (Figure 1) that serve as upper bounds on the desired loss function are often used to create tractable optimization problems. The popular support vector machine (SVM) classifier [Cortes and Vapnik, 1995], for example, employs the hinge-loss—an upper bound on the zero-one loss—to avoid the often intractable empirical risk minimization problem. Adaboost [Freund and Schapire, 1997] incrementally minimizes the exponential loss. The difference between the actual loss and its convex surrogate can introduce a substantial mismatch between optimal parameter estimation under the surrogate loss function and optimal parameter estimates for the original performance objective.

Adaboost [Freund and Schapire, 1997] incrementally minimizes the exponential loss. The difference between the actual loss and its convex surrogate can introduce a substantial mismatch between optimal parameter estimation under the surrogate loss function and optimal parameter estimates for the original performance objective.

### 2.2 COST-SENSITIVE LEARNING

Cost-sensitive learning considers more general loss functions than the zero-one loss in which the loss depends on the actual and the predicted class. One approach is to estimate the conditional label distribution,  $\tilde{P}(y|\mathbf{x})$ , and employ the Bayesian optimal classifier:  $\hat{f}(\mathbf{x}) =$

<sup>1</sup>Lowercase non-bold,  $x$ , and bold,  $\mathbf{x}$ , denote scalar and vector values, and capitals,  $X$  or  $\mathbf{X}$ , denote random variables.

$\operatorname{argmin}_{y' \in \mathcal{Y}} \mathbb{E}_{\tilde{P}(y|\mathbf{x})} [C_{y', Y}]$ , using, e.g., the cost matrix of Eq. (1). However, accurately estimating the conditional label distribution will typically require much more data than methods that directly learn the best class prediction for a given loss function [Margeintu, 2002].

Early meta-learning methods for cost-sensitive learning attempt to modify how a cost-insensitive learner is used during training and/or prediction time so that the end result of its use is cost-sensitive. One approach for this is to either stratify or reweight available training data so that more costly mistakes will incur a larger overall cost and therefore the resulting classifier will be more sensitive to them [Chan and Stolfo, 1998, Elkan, 2001, Zadrozny et al., 2003, Zhou and Liu, 2010]. However, the validity of this approach is limited to a restricted class of *consistent* cost matrices when applied to multi-class prediction tasks [Domingos, 1999, Zhou and Liu, 2010]. A method that reduces multi-class predictions to binary predictions using iterative reweighting, data space expansion, and gradient boosting with stochastic ensembles [Abe et al., 2004] has been proposed to overcome these limitations. The *Metacost* algorithm [Domingos, 1999] similarly wraps around any underlying classifier. It uses bagging to produce label probability estimates, which it then uses to modify training data labels to produce more cost-sensitive predictions on the training set.

Direct cost-sensitive learning methods incorporate the confusion costs directly into the formulation of the classifier. Some classification methods are much more amenable to cost-sensitive modifications than others. In decision trees, for example, modified criteria for greedily selecting decision nodes and/or pruning the tree based on the confusion cost have been successfully employed [Knoll et al., 1994, Turney, 1995, Elkan, 2001, Ling et al., 2004, Davis et al., 2006, Lomax and Vadera, 2013], while relatively little attention has been given for developing cost-sensitive nearest neighbor classifiers [Qin et al., 2013].

Boosting iteratively creates an ensemble of weak classifiers that are then combined to create a much stronger classifier [Freund and Schapire, 1997] that often performs well in practice. Cost-sensitive boosting techniques employ cost-sensitive weak learners to produce a stronger learner that is cost-sensitive as well [Fan et al., 1999, Ting, 2000]. This is accomplished by minimizing the risk over the training dataset,  $\frac{1}{n} \sum_{i=1}^n \operatorname{loss}'(C, y_i, S(\mathbf{x}_i))$ , using a generalized surrogate loss function,  $\operatorname{loss}'(C, \tilde{y}, S_m(\mathbf{x}))$ , for the cost matrix  $C$ , class label  $\tilde{y}$ , and where  $S_y(\mathbf{x})$  represents the classifier confidence in assigning class  $y$  to data point  $\mathbf{x}$ . Recently developed loss functions are the Generalized Exponential Loss (GEL),  $\sum_{y'} C_{y, y'} e^{S_{y'}(\mathbf{x}) - S_y(\mathbf{x})}$  and the Generalized Logistic Loss (GLL),  $\log(1 + \sum_{y'} C_{y, y'} e^{S_{y'}(\mathbf{x}) - S_y(\mathbf{x})})$ . These loss functions are guess-averse and produce state-of-the-art perfor-

mance when used in boosting for cost-sensitive classification [Beijbom et al., 2014].

Support vector machines [Cortes and Vapnik, 1995] have been generalized in the binary classification setting by penalizing mistakes for one class more than for the other class [Brefeld et al., 2003]. Multiclass problems are reduced to binary classifiers using one-versus-all [Bottou et al., 1994] and one-versus-one [Knerr et al., 1990] prediction tasks. The *Cost-Sensitive One-Versus-All* (CSOVA) algorithm [Lin, 2008] trains a separate binary SVM classifier for each class. The *Cost-Sensitive One-Versus-One* (CSOVO) algorithm [Lin, 2010] instead constructs a total of  $k(k-1)/2$  classifiers—one for each pair of classes  $(i, j)$ . For both CSOVA and CSOVO, binary classifiers are aggregated to produce a multi-class prediction. Using structured SVM methods [Tsochantaridis et al., 2005] to directly incorporate cost-sensitivity into the multiclass generalization of the hinge loss [Lee et al., 2004],

$$\min_{\theta, \epsilon \geq 0} \theta \cdot \theta + \alpha \sum_i \epsilon_i \text{ such that:} \quad (3)$$

$$\theta \cdot \phi(\mathbf{x}_i, y_i) - \theta \cdot \phi(\mathbf{x}_i, y') \geq C_{y', y_i} - \epsilon_i, \forall i, y' \neq y_i,$$

creates a margin-based classifier that incorporates mistake costs additively. We note that central to each of these SVM-based methods is the hinge loss approximation of the cost-sensitive loss function. Our approach avoids such approximations of the loss function by instead approximating the available training data.

### 2.3 ADVERSARIAL METHODS

The adversarial perspective that we leverage in our approach has played a formative role in statistical estimation and decision making under uncertainty. These include Wald’s maximin model [Wald, 1949] of decision making as a sequential adversarial game, Savage’s minimax optimization of the regret of decisions [Savage, 1951], and statistical estimates under uncertainty that minimize worst-case risk [Wolfowitz, 1950]. We follow a relaxation of this idea, which estimates complete probability distributions as solutions to a minimax game [Topsøe, 1979, Grünwald and Dawid, 2004]. This formulation is most commonly known as a means for deriving the principle of maximum entropy using the logarithmic loss. From this, many exponential family distributions (e.g., Gaussian distribution, exponential) can be derived [Wainwright and Jordan, 2008].

Our approach differs substantially from adversarial machine learning formulations that are made robust to adversarial shifts in the dataset [Dalvi et al., 2004, Liu and Ziebart, 2014] or uncertainty in the loss function [Wang and Tang, 2012]. We assume training and testing data are IID and the cost-sensitive loss function is fully known. We restrict our uncertainty to the conditional label

distribution  $P(y|\mathbf{x})$  and adversarially estimate it. In contrast with minimax approaches to classification that assume parametric forms of the data [Lanckriet et al., 2003], our approach allows the estimation of any conditional label distribution. Instead, only training data properties are incorporated in the form of constraints on the adversary’s conditional label distribution [Grünwald and Dawid, 2004].

## 3 ADVERSARIAL COST-SENSITIVITY

### 3.1 FORMULATION

We begin to define our notation by considering an estimator for the conditional label distribution,  $\hat{P}(y|\mathbf{x})$ , the actual evaluation distribution  $P(y|\mathbf{x})$ , and an adversarial distribution  $\check{P}(y|\mathbf{x})$ . We compactly represent each as  $|\mathcal{Y}|$ -sized vectors  $\hat{\mathbf{p}}_{\mathbf{x}} = [\hat{P}(\hat{Y} = 1|\mathbf{x}) \hat{P}(\hat{Y} = 2|\mathbf{x}) \dots]^T$  for each input  $\mathbf{x} \in \mathcal{X}$ , and, similarly,  $\mathbf{p}_{\mathbf{x}}$  and  $\check{\mathbf{p}}_{\mathbf{x}}$ . The expected loss suffered from this estimator on input  $x$  for a confusion cost matrix  $\mathbf{C}$  is:  $\hat{\mathbf{p}}_{\mathbf{x}}^T \mathbf{C} \mathbf{p}_{\mathbf{x}} = \mathbb{E}_{\hat{P}(\hat{y}|\mathbf{x})P(y|\mathbf{x})}[C_{\hat{Y}, Y}]$ .

Only samples from the true conditional label distribution  $P(y|\mathbf{x})$  are available. We denote these by distribution  $\check{P}(y|\mathbf{x})$  (compactly represented as  $\check{\mathbf{p}}_{\mathbf{x}}$ ) and also input sample distribution  $\tilde{P}(\mathbf{x})$ . Minimizing the empirical risk under this distribution,  $\mathbb{E}_{\tilde{P}(\mathbf{x})}[\hat{\mathbf{p}}_{\theta, \mathbf{x}}^T \mathbf{C} \check{\mathbf{p}}_{\mathbf{x}}] = \frac{1}{n} \sum_{i=1}^n \sum_{\hat{y} \in \mathcal{Y}} \tilde{P}(\hat{y}|\mathbf{x}_i) C_{\hat{y}, y_i}$ , for some parametric form of the estimation distribution, e.g.,  $\hat{P}_{\theta}(y|\mathbf{x}) \propto e^{\theta \cdot \phi(\mathbf{x}, y)}$ , leads to a non-convex and generally intractable optimization problem, assuming  $\mathbf{P} \neq \mathbf{NP}$ , as discussed in §2.1.

To avoid these non-convex optimization concerns, we employ a robust minimax formulation [Topsøe, 1979, Grünwald and Dawid, 2004] to construct our cost-sensitive classifier (Definition 1). This formulation views the estimation task as a two-player game between estimator and adversary. The adversary is constrained to choose distributions that match a vector of moment statistics of the distribution,  $\mathbb{E}_{P(\mathbf{x})P(y|\mathbf{x})}[\phi(\mathbf{X}, Y)]$ . We denote the set of conditional distributions  $P(y|\mathbf{x})$  satisfying these statistics as  $\Xi$ .

**Definition 1.** *In the constrained cost-sensitive minimax game, the estimator player first selects a predictive distribution,  $\hat{\mathbf{p}}_{\mathbf{x}} \triangleq \hat{P}(\hat{y}|\mathbf{x}) \in \Delta$ , for each input  $\mathbf{x}$ , from the conditional probability simplex  $\Delta$ , and then the adversarial player selects an evaluation distribution,  $\check{\mathbf{p}}_{\mathbf{x}} \triangleq \check{P}(\check{y}|\mathbf{x}) \in \Delta$ , for each input  $x$  from the set  $\Xi$  of distributions consistent with known statistics:*

$$\min_{\{\hat{\mathbf{p}}_{\mathbf{x}}\} \in \Delta} \max_{\{\check{\mathbf{p}}_{\mathbf{x}}\} \in \Xi \cap \Delta} \mathbb{E}_{P(\mathbf{x})}[\hat{\mathbf{p}}_{\mathbf{x}}^T \mathbf{C} \check{\mathbf{p}}_{\mathbf{x}}] \quad (4)$$

where:  $\Xi : \mathbb{E}_{P(\mathbf{x})\check{P}(\check{y}|\mathbf{x})}[\phi(\mathbf{X}, \check{Y})] = \tilde{\phi}$ .

We denote the set of conditional probabilities for each input  $\mathbf{x}$  as  $\{\hat{\mathbf{p}}_{\mathbf{x}}\}$  and  $\{\check{\mathbf{p}}_{\mathbf{x}}\}$ . Here,  $\tilde{\phi}$  is a vector of provided feature moments measured from sample training data,  $\tilde{\phi} = \mathbb{E}_{\tilde{P}(\mathbf{x}, y)}[\phi(\mathbf{X}, Y)]$ , for example.



Conceptually, the feature statistics  $\phi(\mathbf{x}, y)$  defining the set  $\Xi$  should be chosen to restrict the adversary as much as possible from maximizing the loss. However, defining the set to be too restrictive leads to overfitting to the training data. Indeed, the complexity of the estimator  $\hat{P}(\hat{y}|\mathbf{x})$  implicitly grows with the dimensionality of the constraints in  $\Xi$ . Thoughtfully specifying the feature function  $\phi(\cdot, \cdot)$  and employing regularization can avoid this issue (§3.4).

### 3.2 INFERENCE AS ZERO-SUM GAME EQUILIBRIA

We establish efficient inference algorithms for our approach in this section. Theorem 1 transforms the joint adversary-constrained zero-sum games over many different inputs  $\mathbf{x}$  into a set of unconstrained zero-sum game that are independent for each input  $\mathbf{x}$  and connected by a parameterized cost matrix defining each player’s game outcomes.

**Theorem 1.** *Determining the value of the constrained cost-sensitive minimax game reduces to a minimization over the expectation of many unconstrained minimax game:*

$$\min_{\{\hat{\mathbf{p}}_{\mathbf{x}}\} \in \Delta} \max_{\{\check{\mathbf{p}}_{\mathbf{x}}\} \in \Xi \cap \Delta} \mathbb{E}_{P(\mathbf{x})}[\hat{\mathbf{p}}_{\mathbf{x}}^T \mathbf{C} \check{\mathbf{p}}_{\mathbf{x}}] \quad (5)$$

$$= \max_{\{\hat{\mathbf{p}}_{\mathbf{x}}\} \in \Xi \cap \Delta} \mathbb{E}_{P(\mathbf{x})} \left[ \min_{\check{\mathbf{p}}_{\mathbf{x}} \in \Delta} \hat{\mathbf{p}}_{\mathbf{x}}^T \mathbf{C} \check{\mathbf{p}}_{\mathbf{x}} \right] \quad (6)$$

$$= \min_{\theta} \mathbb{E}_{P(\mathbf{x})} \left[ \max_{\check{\mathbf{p}}_{\mathbf{x}} \in \Delta} \min_{\hat{\mathbf{p}}_{\mathbf{x}} \in \Delta} \hat{\mathbf{p}}_{\mathbf{x}}^T \mathbf{C}'_{\mathbf{x}, \theta} \check{\mathbf{p}}_{\mathbf{x}} \right], \quad (7)$$

where  $\theta$  parametrizes the new game characterized by matrix  $\mathbf{C}'_{\mathbf{x}, \theta} : (C'_{\mathbf{x}, \theta})_{\hat{y}, \check{y}} = C_{\hat{y}, \check{y}} + \theta^T (\phi(\mathbf{x}, \hat{y}) - \phi(\mathbf{x}, \check{y}))$ , and  $\phi(\cdot, \cdot)$  terms are from the definition of set  $\Xi$ .

*Proof of Theorem 1.*

$$\begin{aligned} & \min_{\{\hat{\mathbf{p}}_{\mathbf{x}}\} \in \Delta} \max_{\{\check{\mathbf{p}}_{\mathbf{x}}\} \in \Xi \cap \Delta} \mathbb{E}_{P(\mathbf{x})}[\hat{\mathbf{p}}_{\mathbf{x}}^T \mathbf{C} \check{\mathbf{p}}_{\mathbf{x}}] \\ \stackrel{(a)}{=} & \max_{\{\hat{\mathbf{p}}_{\mathbf{x}}\} \in \Xi \cap \Delta} \min_{\check{\mathbf{p}}_{\mathbf{x}} \in \Delta} \mathbb{E}_{P(\mathbf{x})}[\hat{\mathbf{p}}_{\mathbf{x}}^T \mathbf{C} \check{\mathbf{p}}_{\mathbf{x}}] \\ \stackrel{(b)}{=} & \max_{\{\hat{\mathbf{p}}_{\mathbf{x}}\} \in \Xi \cap \Delta} \mathbb{E}_{P(\mathbf{x})} \left[ \min_{\check{\mathbf{p}}_{\mathbf{x}} \in \Delta} \hat{\mathbf{p}}_{\mathbf{x}}^T \mathbf{C} \check{\mathbf{p}}_{\mathbf{x}} \right] \\ \stackrel{(c)}{=} & \max_{\{\hat{\mathbf{p}}_{\mathbf{x}}\} \in \Delta} \min_{\theta} \mathbb{E}_{P(\mathbf{x})} \left[ \min_{\check{\mathbf{p}}_{\mathbf{x}} \in \Delta} \hat{\mathbf{p}}_{\mathbf{x}}^T \mathbf{C} \check{\mathbf{p}}_{\mathbf{x}} \right. \\ & \quad \left. + \theta^T \mathbb{E}_{P(\mathbf{x})}[\Phi_{\mathbf{x}}(\check{\mathbf{p}}_{\mathbf{x}} - \hat{\mathbf{p}}_{\mathbf{x}})] \right] \\ \stackrel{(d)}{=} & \min_{\theta} \mathbb{E}_{P(\mathbf{x})} \left[ \max_{\check{\mathbf{p}}_{\mathbf{x}} \in \Delta} \min_{\hat{\mathbf{p}}_{\mathbf{x}} \in \Delta} \hat{\mathbf{p}}_{\mathbf{x}}^T \mathbf{C}'_{\mathbf{x}, \theta} \check{\mathbf{p}}_{\mathbf{x}} \right] \end{aligned}$$

where  $\Phi$  is the matrix defined by  $\Phi_{i,j} = \phi_i(\mathbf{x}, y_j)$  and  $\mathbf{C}'_{\mathbf{x}}$  is defined by elements:

$$(C'_{\mathbf{x}})_{\hat{y}, \check{y}} = C_{\hat{y}, \check{y}} + \theta^T (\phi(\mathbf{x}, \hat{y}) - \phi(\mathbf{x}, \check{y})). \quad (8)$$

Step (a) follows from minimax duality in zero-sum games [von Neumann and Morgenstern, 1947]. As an affine function of terms each with individual  $\check{\mathbf{p}}_{\mathbf{x}}$  term,

each minimization can be performed independently in step (b). Step (c) expresses the primal Lagrangian. For step (d),  $\mathbb{E}_{P(\mathbf{x})}[\min_{\check{\mathbf{p}}_{\mathbf{x}} \in \Delta} \hat{\mathbf{p}}_{\mathbf{x}}^T \mathbf{C} \check{\mathbf{p}}_{\mathbf{x}} + \theta^T \Phi_{\mathbf{x}}(\check{\mathbf{p}}_{\mathbf{x}} - \hat{\mathbf{p}}_{\mathbf{x}})]$ —a non-negative linear combination of minimums of affine functions—is a concave function of  $\check{\mathbf{p}}_{\mathbf{x}}$  terms. Given a feasible solution on the relative interior of  $\Xi$  [Boyd and Vandenberghe, 2004], strong Lagrangian duality holds. As in step (b), the maximizations can then be independently applied.  $\square$

Figure 2 shows the value of the game for a single  $\mathbf{x}$  from Eq. (6) as a function of the adversarial distribution  $\check{\mathbf{p}}_{\mathbf{x}}$  for zero-one loss and a more general cost matrix. The adversary is not free to independently maximize these functions for each  $\mathbf{x}$ , but must instead choose a structured prediction that resides within the constraint set  $\Xi$ .

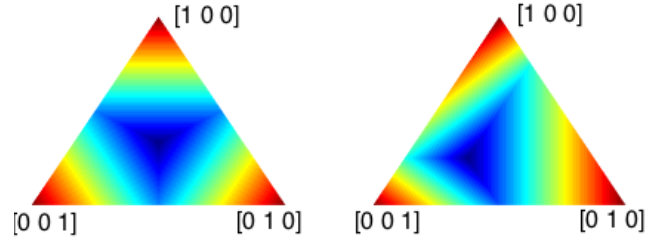


Figure 2: The portion of the adversary’s objective function (6) for a single example,  $\min_{\check{\mathbf{p}}_{\mathbf{x}} \in \Delta} \hat{\mathbf{p}}_{\mathbf{x}}^T \mathbf{C} \check{\mathbf{p}}_{\mathbf{x}}$ , in the adversary-constrained game for zero-one loss (left) and a more general cost-sensitive loss with cost matrix  $[0 \ 2 \ 3; 2 \ 0 \ 1; 1 \ 3 \ 0]$  (right) in a three-class prediction task.

After applying Theorem 1 and given model parameters,  $\theta$ , (obtaining these parameters is discussed in §3.3) the unconstrained game,  $\max_{\check{\mathbf{p}}_{\mathbf{x}} \in \Delta} \min_{\hat{\mathbf{p}}_{\mathbf{x}} \in \Delta} \hat{\mathbf{p}}_{\mathbf{x}}^T \mathbf{C}'_{\mathbf{x}, \theta} \check{\mathbf{p}}_{\mathbf{x}}$ , can be solved independently for each  $\mathbf{x}$ . In this augmented game, our original cost matrix from Eq. (1) is transformed into the augmented cost matrix:

$$\mathbf{C}' = \begin{bmatrix} 0 + \psi_1 & 1 + \psi_2 & 2 + \psi_3 & 0 + \psi_4 \\ 3 + \psi_1 & 0 + \psi_2 & 1 + \psi_3 & 3 + \psi_4 \\ 4 + \psi_1 & 2 + \psi_2 & 0 + \psi_3 & 1 + \psi_4 \\ 1 + \psi_1 & 1 + \psi_2 & 2 + \psi_3 & 0 + \psi_4 \end{bmatrix}, \quad (9)$$

where Lagrangian potentials are compactly denoted as  $\psi_i = \theta^T (\phi(\mathbf{x}, i) - \phi(\mathbf{x}, \tilde{y}))$  with  $\tilde{y}$  representing the example’s actual label. For parameter estimation, the second feature function based on the actual label  $\tilde{y}$  serves an important role. However, since it is constant with respect to  $\tilde{y}$  and  $\hat{y}$ , and therefore does not influence the solution strategies for the game, it can be ignored when making predictions on data with unknown labels (or assigned an arbitrary value from  $\mathcal{Y}$  without affecting predictions).

Figure 3 shows the adversary’s objective function in the unconstrained, cost-augmented game. Conceptually, the adversary’s objective function from the constrained game

(Figure 2) is “placed” on top of a hyperplane shaped by the Lagrangian potential terms,  $\psi_i$ . The difference in these potential terms determines the adversary’s equilibrium strategy. For the binary classification task, there are three possible equilibrium strategies for the adversary. With three classes, there are seven possibilities: three pure strategies; three strategies that are mixtures of two classes; and one strategy that is a mixture of all three classes.

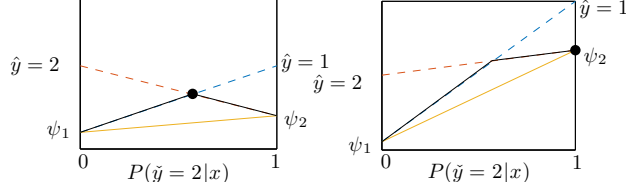


Figure 3: The adversary’s objective in the unconstrained game for a binary classification task with a mixed (uncertain) equilibrium solution (left) and a pure (certain) equilibrium solution (right). The third adversary strategy,  $P(\hat{y} = 2|x) = 0$ , is realized when  $\psi_1 \gg \psi_2$ .

Unlike the logarithmic loss under this minimax formulation, the cost-sensitive loss function does not provide a closed-form parametric solution<sup>2</sup>. Instead, the inner minimax game (inside the expectation of Eq. (7)) for each input  $\mathbf{x}$  can be solved as a linear program [von Neumann and Morgenstern, 1947]:

$$\begin{aligned} & \max_{v, \tilde{P}(\tilde{y}|\mathbf{x})} v & (10) \\ \text{subject to: } & v \leq \sum_{\tilde{y} \in \mathcal{Y}} \tilde{P}(\tilde{y}|\mathbf{x}) (C'_{\mathbf{x},\theta})_{\tilde{y},\tilde{y}} \quad \forall \tilde{y} \in \mathcal{Y} \\ & \sum_{\tilde{y} \in \mathcal{Y}} \tilde{P}(\tilde{y}|\mathbf{x}) = 1 \text{ and } \tilde{P}(\tilde{y}|\mathbf{x}) \geq 0, \quad \forall \tilde{y} \in \mathcal{Y}. \end{aligned}$$

The resulting distribution,  $\tilde{P}(\tilde{y}|\mathbf{x})$ , gives the adversary’s strategy  $\tilde{\mathbf{p}}_{\mathbf{x}}^*$ . The other strategy of the Nash equilibrium strategy pair,  $(\tilde{\mathbf{p}}_{\mathbf{x}}^*, \hat{\mathbf{p}}_{\mathbf{x}}^*)$  can be obtained by solving the same linear program with the cost matrix transposed and negated.

### 3.3 LEARNING VIA CONVEX OPTIMIZATION

Our key remaining task for employing the proposed approach is to obtain model parameters (Lagrangian multipliers)  $\theta$  that enforce the adversarial distribution to reside within the constraint set  $\Xi$ .

**Theorem 2.** *The subdifferential of the outer minimization problem (Eq. (7)) includes the expected feature difference as a subgradient:*

$$\begin{aligned} & \mathbb{E}_{P(\mathbf{x})\tilde{P}_{\hat{\theta}}^*(\tilde{y}|\mathbf{x})} [\phi(\mathbf{X}, \tilde{Y})] - \mathbb{E}_{P(\mathbf{x})P(y|\mathbf{x})} [\phi(\mathbf{X}, Y)] & (11) \\ & \in \partial_{\theta} \mathbb{E}_{P(\mathbf{x})} \left[ \min_{\tilde{\mathbf{p}}_{\mathbf{x}} \in \Delta} \max_{\hat{\mathbf{p}}_{\mathbf{x}} \in \Delta} \hat{\mathbf{p}}_{\mathbf{x}}^T C'_{\mathbf{x},\theta} \tilde{\mathbf{p}}_{\mathbf{x}} \right] \Big|_{\theta=\hat{\theta}} \end{aligned}$$

<sup>2</sup>Adversarial logarithmic loss minimization yields members of the exponential family [Wainwright and Jordan, 2008].

where  $\tilde{P}^*(\tilde{y}|\mathbf{x})$  is the solution to Eq. (10).

*Proof of Theorem 2.* Taking the subdifferential, we have:

$$\begin{aligned} & \partial_{\theta_k} \mathbb{E}_{P(\mathbf{x})} \left[ \min_{\tilde{\mathbf{p}}_{\mathbf{x}} \in \Delta} \max_{\hat{\mathbf{p}}_{\mathbf{x}} \in \Delta} \hat{\mathbf{p}}_{\mathbf{x}}^T C'_{\mathbf{x},\theta} \tilde{\mathbf{p}}_{\mathbf{x}} \right] \Big|_{\theta=\hat{\theta}} \\ & \stackrel{(a)}{=} \mathbb{E}_{P(\mathbf{x})} \left[ \partial_{\theta_k} \max_{\tilde{\mathbf{p}}_{\mathbf{x}} \in \Delta} \min_{\hat{\mathbf{p}}_{\mathbf{x}} \in \Delta} \hat{\mathbf{p}}_{\mathbf{x}}^T C'_{\mathbf{x},\theta} \tilde{\mathbf{p}}_{\mathbf{x}} \right] \Big|_{\theta=\hat{\theta}} \\ & \stackrel{(b)}{\ni} \mathbb{E}_{P(\mathbf{x})} \left[ \partial_{\theta_k} (\hat{\mathbf{p}}_{\mathbf{x}}^*)^T C'_{\mathbf{x},\theta} \tilde{\mathbf{p}}_{\mathbf{x}}^* \right] \Big|_{\theta=\hat{\theta}} \\ & \stackrel{(c)}{=} \mathbb{E}_{P(\mathbf{x})} \left[ (\hat{\mathbf{p}}_{\mathbf{x}}^*)^T (\partial_{\theta_k} C'_{\mathbf{x},\theta}) \tilde{\mathbf{p}}_{\mathbf{x}}^* \right] \Big|_{\theta=\hat{\theta}} \\ & \stackrel{(d)}{\ni} \mathbb{E}_{P(\mathbf{x})\tilde{P}_{\hat{\theta}}^*(\tilde{y}|\mathbf{x})} [\phi_k(\mathbf{X}, \tilde{Y})] - \mathbb{E}_{P(\mathbf{x})P(y|\mathbf{x})} [\phi_k(\mathbf{X}, Y)]. \end{aligned}$$

Step (a) follows from the rule for non-negative combinations of subdifferentials. Step (b) follows from the subdifferential of the function evaluated at the maximizing/minimizing values being a subset of the subdifferential of the maximum/minimum functions. Step (c), like step (a), follows from the rule for non-negative combinations of subdifferentials by noting that  $(\hat{\mathbf{p}}_{\mathbf{x}}^*)^T C'_{\mathbf{x},\theta} \tilde{\mathbf{p}}_{\mathbf{x}}^* = \hat{\mathbf{p}}_{\mathbf{x}}^* (\tilde{\mathbf{p}}_{\mathbf{x}}^*)^T \bullet C'_{\mathbf{x},\theta}$ , where  $\bullet$  represents the “matrix dot product” (i.e.,  $\mathbf{A} \bullet \mathbf{B} \triangleq \sum_{i,j} A_{i,j} B_{i,j}$ ). In step (d), the subdifferential terms for  $C'_{\mathbf{x}}$  include  $\phi_k(\mathbf{x}, \tilde{y}) - \phi_k(\mathbf{x}, y) \in (\partial_{\theta_k} C'_{\mathbf{x}})_{\tilde{y},y}$  and do not depend on  $\hat{\mathbf{p}}_{\mathbf{x}}$ .  $\square$

Leveraging the convexity of the formulation’s objective function (discussed in the Proof of Theorem 1), and using the common substitution of the sample training data distribution,  $\tilde{P}(\mathbf{x})$ , in place of the distribution  $P(\mathbf{x})$ , we employ standard subgradient-based optimization methods for convex optimization problems to obtain parameters for our cost-sensitive classifier (Algorithm 1).

---

#### Algorithm 1 Parameter estimation for the robust cost-sensitive classifier

---

**Input:** Cost matrix  $\mathbf{C}$ , training dataset  $\mathcal{D}$  with pairs  $(\tilde{\mathbf{x}}_i, \tilde{y}_i) \in \mathcal{D}$ , feature function  $\phi : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}^k$ , time-varying learning rate  $\{\gamma_t\}$

**Output:** Model parameter estimate  $\theta$

```

t ← 1
while θ not converged do
  for all (x-tilde_i, y-tilde_i) ∈ D do
    Construct cost matrix C'_x-tilde_i,θ using Eq. (8)
    Solve for P-tilde(y|x-tilde_i) using the LP of Eq. (10)
    ∇_θ = E_{P-tilde(y|x-tilde_i)} [φ(x-tilde_i, Y)] - φ(x-tilde_i, y-tilde_i)
    θ = θ - γ_t ∇_θ
  t ← t + 1
end for
end while

```

---

Though we describe a stochastic subgradient in our algorithm, any convex optimization method for non-smooth objective functions can be employed.

### 3.4 PERFORMANCE GUARANTEES & ILLUSTRATIVE EXAMPLES

We establish performance guarantees and illustrate the behavior of our approach in this portion of the paper. We focus specifically on the similarities to and differences from support vector machines [Cortes and Vapnik, 1995] and their structured extensions [Tsochantaridis et al., 2004]. Given ideal data (linearly separable), Theorem 3 establishes an equivalence to hard-margin SVMs.

**Theorem 3.** *Given linearly separable training data, i.e.,*

$$\exists \theta : \forall i, y' \neq y_i, \theta \cdot \phi(\mathbf{x}_i, y_i) > \theta \cdot \phi(\mathbf{x}_i, y'), \quad (12)$$

and zero cost only for correct predictions  $C_{i,i} = 0$ , the adversarial cost-sensitive learner with sufficiently small  $L_2$  regularization is equivalent to a hard-margin cost-sensitive support vector machine.

*Proof.* Eq. (12) implies  $\exists \theta' : \forall i, y' \neq y_i, \theta' \cdot \phi(\mathbf{x}_i, y_i) > \theta' \cdot \phi(\mathbf{x}_i, y') + C_{y', y_i}$  (the hard-margin cost-sensitive SVM constraint set with  $\epsilon = \mathbf{0}$  in Eq. (3)) by multiplicatively scaling  $\theta$ . The Nash equilibrium is  $\hat{P}(\hat{y}_i | \mathbf{x}_i) = 1$  and  $\hat{P}(\hat{y}_i | \mathbf{x}_i) = 1$  with a cost-sensitive loss of zero *if and only if* this inequality is satisfied. Given this, the dual optimization in Eq. (7) realizes its minima (zero loss) only when these constraints are satisfied. The  $L_2$  regularization term is a monotonic transformation of the objective of the hard-margin SVM:  $\theta \cdot \theta$ . Thus, having the same constraints and objective functions with corresponding maxima, an equivalent solution is produced.  $\square$

As a result of this equivalence to hard-margin SVM, adversarial classification inherits the convergence properties of support vectors machines in the realizable case of Eq. (12).

The game strategies of each player are illustrated in Figure 4 for binary prediction using the zero-one loss in the separable setting. Between perfectly classified datapoints, our approach produces a region of uncertainty that is maximally uncertain for the adversary’s Nash equilibrium strategy ( $\hat{P}(\hat{Y} = \text{‘o’} | \mathbf{x}) = 0.5$ ), while the predictor’s Nash equilibrium strategy smoothly transitions from one class to the other in this region.

Given non-separable data, the adversarial approach suggests choosing a set  $\Xi$  of constraints based on training samples  $\hat{P}(\mathbf{x}, y)$  that will also contain the true label distribution,  $P(y | \mathbf{x})$ . When this is accomplished, Theorem 4 provides performance guarantees for generalization.

**Theorem 4.** *If  $P(y | \mathbf{x}) \in \Xi$ , confusion costs from the adversarial game upper bound the generalization error confusion costs:*

$$\mathbb{E}_{P(\mathbf{x})P(y|\mathbf{x})\hat{P}^*(\hat{y}|\mathbf{x})}[C_{\hat{Y}, Y}] \leq \mathbb{E}_{P(\mathbf{x})\hat{P}^*(\hat{y}|\mathbf{x})\hat{P}^*(\hat{y}|\mathbf{x})}[C_{\hat{Y}, \hat{Y}}].$$

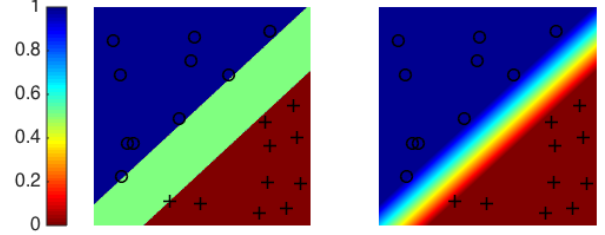


Figure 4: Adversary (left) and predictor (right) distributions for separable data under zero-one loss

*Proof.* By definition, the adversarial conditional label distribution,  $\hat{P}^*(\hat{y} | \mathbf{x})$ , is a Nash equilibrium and it provides the worst possible loss for the estimator of all conditional label distributions from set  $\Xi$ . So long as the true label distribution used for evaluation,  $P(y | \mathbf{x})$ , is similar to training data properties (i.e, a member of  $\Xi$ ), then costs that are no worse than  $\hat{P}^*(\hat{y} | \mathbf{x})$  can result without  $P(y | \mathbf{x})$  being a better choice from  $\Xi$  than  $P(y | \mathbf{x})$  for maximizing the predictor’s loss, a contradiction.  $\square$

Slack can be added to the constraint set  $\Xi$  or regularization to the dual optimization problem of Eq. (6) to address finite sample approximation error when using sample data,  $\mathbb{E}_{\hat{P}(\mathbf{x}, y)}[\phi(\mathbf{X}, Y)]$ , as an estimate of the distribution’s statistics,  $\mathbb{E}_{P(\mathbf{x}, y)}[\phi(\mathbf{X}, Y)]$ .

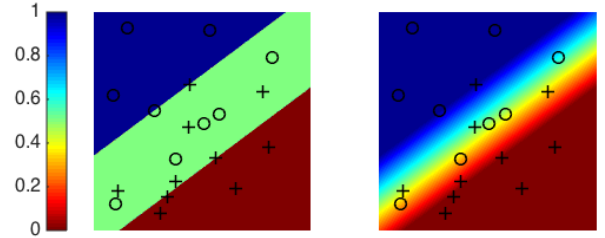


Figure 5: Adversary (left) and predictor (right) distributions for nonseparable data under zero-one loss

Figure 5 shows the two equilibria strategies for data that is not linearly separable in the zero-one loss binary classification setting. The uncertainty region of our approach depends on summary statistics rather than the specific datapoint labels that define margin boundaries of SVMs (Figure 5). Increased non-separability of the data and greater regularization amounts expand this uncertainty region.

The equilibria under cost-sensitive losses, shown in Figure 6 shifts the region of uncertainty to better minimize the expected cost compared to Figure 5, which is based on the same data sample. Additionally, the adversary’s predictions shift ( $\hat{P}(Y = \text{‘o’} | \mathbf{x}) = .25$ ) within the region of uncertainty.

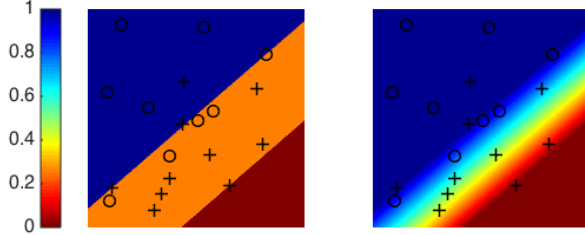


Figure 6: Adversary (left) and predictor (right) distributions for nonseparable data under  $[0 \ 1; 3 \ 0]$  cost matrix.

From the perspective of Theorem 3 and Theorem 4, adversarial cost-sensitive classification provides an alternative to hinge-loss “softening” of the hard-margin SVM. By posing cost-sensitive prediction as an adversarial game (Def. 1), our approach approximates aspects of the training data while being able to employ non-convex loss functions without the intractability encountered by empirical risk minimization. Prediction under this approach reduces to the well-studied problem of solving a zero-sum game, which is easily addressed using linear programming via Eq. (10). This is only a little more complicated than predictions for SVM based on the label that maximizes a linear potential function. Like SVMs, estimating model parameters can be posed as a convex optimization problem and solved using subgradient optimization methods (Alg. 1) under our approach.

## 4 EXPERIMENTS

Our adversarial approach provides the advantage of operating efficiently on non-convex cost-sensitive loss functions, but only through approximating the training data label information rather than minimizing loss on the actual labeled training data. We experimentally investigate the trade-off our approach provides in this section.

### 4.1 DATASETS

We employ publicly available datasets for multiclass classification to evaluate our approach. The number of classes and the number of examples (size) of each dataset are listed in Table 1. We rescale the attributes to  $[0,1]$  and enumerate the class labels.

### 4.2 METHODOLOGY

We conduct 10 cost-sensitive classification tasks for each dataset. We generate confusion cost matrices,  $\mathbf{C}$ , for each task by: (1) assigning all correct classifications a cost of zero ( $C_{i,i} = 0, \forall i$ ); and (2) sampling the remaining elements of the cost matrix from the uniform distribution ( $C_{i,j} \sim U[0, 1], \forall i \neq j$ ). For each classification task, we

Table 1: Evaluation datasets and dataset characteristics.

| Name            | Classes | Attributes | Training | Testing |
|-----------------|---------|------------|----------|---------|
| Iris            | 3       | 4          | 120      | 30      |
| Optical Digits  | 10      | 64         | 3823     | 1797    |
| Satellite Image | 6       | 36         | 4435     | 2000    |
| Shuttle         | 7       | 9          | 43500    | 14500   |
| Vehicle         | 4       | 18         | 658      | 188     |
| Wine            | 3       | 4          | 142      | 36      |
| Breast Tissue   | 6       | 9          | 85       | 21      |
| Ecoli           | 8       | 7          | 269      | 67      |
| Glass           | 6       | 9          | 171      | 43      |
| Image Segment   | 7       | 19         | 210      | 2100    |
| Libras          | 15      | 90         | 288      | 72      |
| Pen Digits      | 10      | 16         | 7494     | 3498    |
| Vertebral       | 3       | 6          | 248      | 62      |

split the data into training and testing sets as described in Table 1. We measure the expected cost of each method averaged over each of the 10 tasks.

### 4.3 COMPARISON METHODS

Our primary points of comparison for investigating this paper’s central hypothesis—that adversarial data approximation produces better cost-sensitive classifiers than convex loss approximation—are support vector methods. However, we also compare with recently reported state-of-the-art cost-sensitive boosting methods. We implement and compare our proposed approach against the following specific methods for cost-sensitive learning. The methodological details for each approach are:

- **Our approach:** We train our method via Algorithm 1 using a quadratic expansion of the original attributes and a “one-hot” encoding of the class label,  $\phi(\mathbf{x}, y) = [\text{vector}(\mathbf{x}\mathbf{x}^T)I(y = 1); \text{vector}(\mathbf{x}\mathbf{x}^T)I(y = 2); \dots]$ . To produce deterministic predictions, we “round” the estimator’s Nash equilibrium strategy,  $\hat{P}^*(\hat{y}|\mathbf{x})$  to the most probable label. This avoids the ambiguity of other methods for making deterministic predictions from mixed strategies (e.g., two or more actions may be the best response to the adversary’s Nash equilibrium strategy).
- **Guess Averse Cost-Sensitive Boosting:** We employ the guess averse cost-sensitive boosting method and implementation [Beijbom et al., 2014] with GLL loss described in §2.1. (We also investigated GEL, but found it to be consistently and significantly outperformed by GLL.) We use a linear regression model as the weak learner.
- **Cost-Sensitive One-Versus-One (CSOVO):** We employ the LIBSVM [Chang and Lin, 2011] implementation of the CSOVO SVM approach [Lin, 2010] described in §2.1. Our experiments use quadratic kernels

Table 2: CSOVO and CSOVA kernel parameters chosen using five-fold cross validation on the training set from  $\gamma_1 \in \{0.125, 1, 2, 5, 10, 1/\text{number of features}\}$  and  $\gamma_0 \in \{1, 2, 5, 10, 50, 100, 200, 300, \dots, 900\}$ .

| Name            | CSOVO      |            | CSOVA      |            |
|-----------------|------------|------------|------------|------------|
|                 | $\gamma_1$ | $\gamma_0$ | $\gamma_1$ | $\gamma_0$ |
| Iris            | 5          | 2          | 1          | 700        |
| Optical Digits  | 1          | 2          | 5          | 2          |
| Satellite Image | 10         | 50         | 1          | 1          |
| Shuttle         | 0.125      | 900        | 0.125      | 900        |
| Vehicle         | 10         | 5          | 10         | 10         |
| Wine            | 1          | 500        | 1          | 5          |
| Breast Tissue   | 0.125      | 900        | 10         | 400        |
| Ecoli           | 5          | 500        | 0.125      | 800        |
| Glass           | 5          | 400        | 10         | 700        |
| Image Segment   | 0.125      | 300        | 0.125      | 600        |
| Libras          | 1          | 5          | 1          | 2          |
| Pen Digits      | 0.125      | 700        | 5          | 5          |
| Vertebral       | 0.125      | 600        | 0.125      | 500        |

[Chang and Lin, 2011],  $K(u, v) = (\gamma_1 u^T v + \gamma_0)^2$  to match the expressiveness of our approach. We run five-fold cross validation on the training set of every dataset to choose quadratic kernel parameters (shown in Table 2), and then we use these best parameters to train from the training set and construct the final classifier model<sup>3</sup> Finally, we evaluate the CSOVO performance by measuring the prediction cost on the test data.

- **Cost-Sensitive One-Versus-All (CSOVA):** We similarly employ the LIBSVM implementation of the CSOVA SVM approach described in §2.1. Our methodology matches that of CSOVO for cross-validation (parameters shown in Table 2), training, and testing.
- **Structured SVM (SVM-Struct):** We employ the Large Scale Structured SVM (SVM LS) software package [Branson et al., 2013] to obtain a multiclass cost-sensitive predictor based on the additive cost-sensitive hinge loss of Eq. (3). SVM LS applies online subgradient methods [Ratliff et al., 2007] and sequential order optimization [Shalev-Shwartz et al., 2011] to improve efficiency. We evaluate the Online Dual Ascent (ODA) algorithm [Branson et al., 2013] as well as the Stochastic Gradient Descent (SGD) method for the purpose of our cost-sensitive experiments. We employ a trade-off parameter  $\alpha$  of 100.

<sup>3</sup>We use the default tolerance of termination criterion, 0.001, for most of the datasets except *image segmentation* and *shuttle*, which required a less sensitive criterion to converge.

## 4.4 RESULTS

Figure 7 shows the average loss incurred by each approach on the 13 different datasets. Our method generally performs well on all of the datasets except *wine* and *libras* datasets and has a similar performance with boosting. SVM methods except SVM-CSOVA are strong on some of the datasets (*optdigits*, *pendigits*, *wine* and *libras*). For many datasets, the performance of the reduction-based SVM approaches is significantly worse than our approach and boosting and the multi-class structured SVM approach. The multi-class structured SVM approach specifically is significantly worse than our method on many of the datasets (*satimage*, *shuttle*, *vehicle*, *breast tissue*, *pendigits*, and *vertebral*), while only significantly better on the *optdigits* dataset.

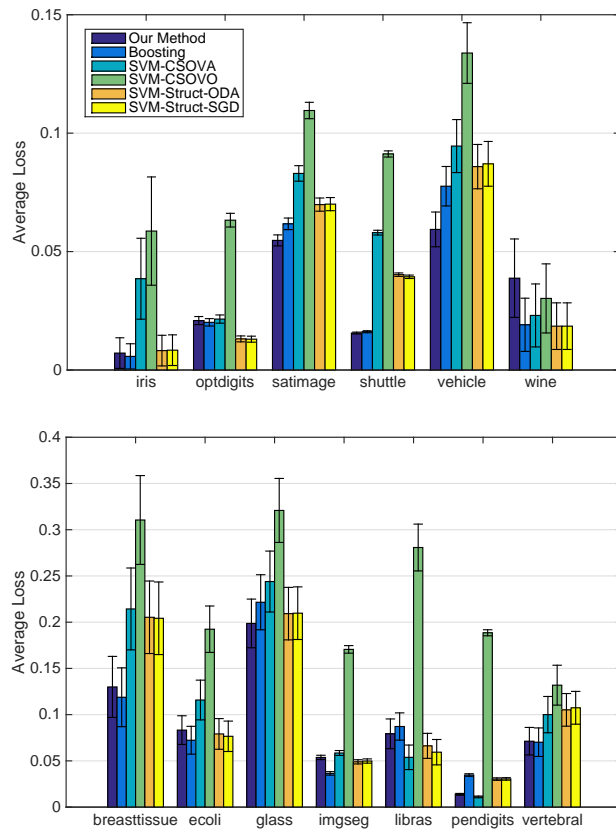


Figure 7: The average loss of predictions for the datasets of Table 1.

The differences between the results of our method and those of boosting are not as extreme. Indeed, for many of the datasets (*iris*, *wine*, *shuttle*, *optdigits*, *vertebral*, *ecoli*, *breast tissue*, and *libras*), the differences in average performance are not significant. For one dataset (*imgseg*), boosting is significantly better, while our method is significantly better for the remaining four (*satimage*, *shuttle*, *vehicle*, and *pendigits*).

We compare the average loss of the prediction methods aggregated over all of the datasets in Figure 8, showing that on average our method provides lower cost predictions. It is important to note that as an ensemble method, boosting is able to implicitly consider a much richer feature space than our approach. For classification, SVMs are often only comparable when incorporating kernels that can also implicitly consider richer feature spaces. Thus, exceeding the performance of the state-of-the-art boosting method using only quadratic features is a significant demonstration of our method. The comparisons with the structured SVM method, which considers an identical feature space, illustrates the general benefit our approach provides by adversarially approximating the training data rather than convexly approximating the loss function.

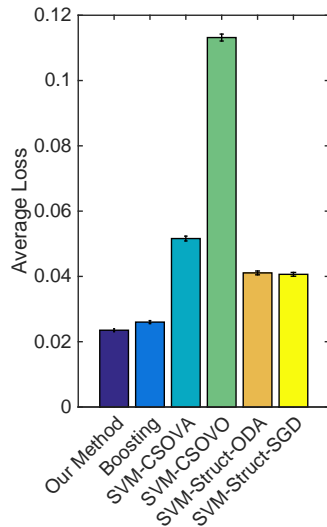


Figure 8: Average loss of predictions across all datasets of Table 1.

## 5 CONCLUSIONS

In this paper, we have developed an approach for minimizing the exact cost-sensitive loss using an adversarial formulation. In stark contrast with existing methods, which typically minimize a convex approximation of the cost-sensitive loss evaluated on available training data, our approach directly minimizes the actual cost-sensitive loss evaluated on an approximation of the training data. This perspective of placing uncertainty around the training data and resolving it by considering an adversarial evaluator leads to a zero-sum game formulation for inference and convex optimization for estimating model parameters.

We demonstrated the benefits of the approach on a total of 130 prediction tasks. Our approach performs competitively with a state-of-the-art boosting method across many of these tasks and better on average. This is despite the fact that boosting, as an ensemble method, is able to implicitly consider a richer feature space for the classifiers that it ultimately produces. The performance of our approach is much more significantly better than structured multi-class SVM methods and reduction-based SVM methods, which are more directly comparable as they employ the same quadratic feature space.

Our future work will investigate avenues for improving and expanding this adversarial approach to cost-sensitive learning. Foremost, we plan investigate the feasibility of incorporating kernel methods with our approach so that much larger or infinite feature spaces can be tractably incorporated into our cost-sensitive classifier. Additionally, we plan to investigate settings with cost functions that depend on the input attributes in addition to the predicted and actual labels.

## Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. #1227495, *Purposeful Prediction: Co-robot Interaction via Understanding Intent and Goals*.

## References

- [Abe et al., 2004] Abe, N., Zadrozny, B., and Langford, J. (2004). An iterative method for multi-class cost-sensitive learning. In *KDD*, pages 3–11. ACM.
- [Beijbom et al., 2014] Beijbom, O., Saberian, M., Kriegman, D., and Vasconcelos, N. (2014). Guess-averse loss functions for cost-sensitive multiclass boosting. In *Proc. International Conference on Machine Learning*, pages 586–594.
- [Bottou et al., 1994] Bottou, L., Cortes, C., Denker, J. S., Drucker, H., Guyon, I., Jackel, L. D., LeCun, Y., Muller, U. A., Sackinger, E., Simard, P., and Vapnik, V. N. (1994). Comparison of classifier methods: a case study in handwritten digit recognition. In *International Conference on Pattern Recognition*, pages 77–82.
- [Boyd and Vandenberghe, 2004] Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press.
- [Branson et al., 2013] Branson, S., Beijbom, O., and Belongie, S. (2013). Efficient large-scale structured learning. In *Computer Vision and Pattern Recognition*, pages 1806–1813. IEEE.
- [Brefeld et al., 2003] Brefeld, U., Geibel, P., and Wyszotzki, F. (2003). Support vector machines with example dependent costs. In *ECML*, pages 23–34. Springer.
- [Chan and Stolfo, 1998] Chan, P. K. and Stolfo, S. J. (1998). Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection. In *KDD*, pages 164–168.
- [Chang and Lin, 2011] Chang, C.-C. and Lin, C.-J. (2011). Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):1–27.
- [Cortes and Vapnik, 1995] Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3):273–297.
- [Dalvi et al., 2004] Dalvi, N., Domingos, P., Sanghai, S., Verma, D., et al. (2004). Adversarial classification. In *KDD*, pages 99–108. ACM.
- [Davis et al., 2006] Davis, J. V., Ha, J., Rossbach, C. J., Ramadan, H. E., and Witchel, E. (2006). Cost-sensitive decision tree learning for forensic classification. In *ECML*, pages 622–629. Springer.

- [Domingos, 1999] Domingos, P. (1999). Metacost: A general method for making classifiers cost-sensitive. In *KDD*, pages 155–164. ACM.
- [Elkan, 2001] Elkan, C. (2001). The foundations of cost-sensitive learning. In *IJCAI*, pages 973–978.
- [Fan et al., 1999] Fan, W., Stolfo, S. J., Zhang, J., and Chan, P. K. (1999). Adacost: misclassification cost-sensitive boosting. In *ICML*, pages 97–105.
- [Freund and Schapire, 1997] Freund, Y. and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139.
- [Grünwald and Dawid, 2004] Grünwald, P. D. and Dawid, A. P. (2004). Game theory, maximum entropy, minimum discrepancy, and robust Bayesian decision theory. *Annals of Statistics*, 32:1367–1433.
- [Hoffgen et al., 1995] Hoffgen, K.-U., Simon, H.-U., and Vanhorn, K. S. (1995). Robust trainability of single neurons. *Journal of Computer and System Sciences*, 50(1):114–125.
- [Knerr et al., 1990] Knerr, S., Personnaz, L., and Dreyfus, G. (1990). Single-layer learning revisited: a stepwise procedure for building and training a neural network. In *Neurocomputing*, pages 41–50. Springer.
- [Knoll et al., 1994] Knoll, U., Nakhaeizadeh, G., and Tausend, B. (1994). Cost-sensitive pruning of decision trees. In *ECML*, pages 383–386. Springer.
- [Lanckriet et al., 2003] Lanckriet, G. R., Ghaoui, L. E., Bhat-tacharyya, C., and Jordan, M. I. (2003). A robust minimax approach to classification. *JMLR*, 3:555–582.
- [Lee et al., 2004] Lee, Y., Lin, Y., and Wahba, G. (2004). Multicategory support vector machines: Theory and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association*, 99(465):67–81.
- [Lin, 2008] Lin, H.-T. (2008). *From ordinal ranking to binary classification*. PhD thesis, California Institute of Technology.
- [Lin, 2010] Lin, H.-T. (2010). A simple cost-sensitive multi-class classification algorithm using one-versus-one comparisons. *National Taiwan University, Tech. Rep.*
- [Ling et al., 2004] Ling, C. X., Yang, Q., Wang, J., and Zhang, S. (2004). Decision trees with minimal costs. In *ICML*, pages 544–551. ACM.
- [Liu and Ziebart, 2014] Liu, A. and Ziebart, B. D. (2014). Robust classification under sample selection bias. In *Advances in Neural Information Processing Systems*, pages 37–45.
- [Lomax and Vadera, 2013] Lomax, S. and Vadera, S. (2013). A survey of cost-sensitive decision tree induction algorithms. *ACM Computing Surveys*, 45(2):16.
- [Margineantu, 2002] Margineantu, D. D. (2002). Class probability estimation and cost-sensitive classification decisions. In *ECML*, pages 270–281. Springer.
- [Qin et al., 2013] Qin, Z., Wang, A. T., Zhang, C., and Zhang, S. (2013). Cost-sensitive classification with k-nearest neighbors. In *Knowledge Science, Engineering and Management*, pages 112–131. Springer.
- [Ratliff et al., 2007] Ratliff, N. D., Bagnell, J. A., and Zinkevich, M. (2007). (approximate) subgradient methods for structured prediction. In *AISTATS*, pages 380–387.
- [Savage, 1951] Savage, L. J. (1951). The theory of statistical decision. *Journal of the American Statistical association*, 46(253):55–67.
- [Shalev-Shwartz et al., 2011] Shalev-Shwartz, S., Singer, Y., Srebro, N., and Cotter, A. (2011). Pegasos: Primal estimated sub-gradient solver for SVM. *Mathematical programming*, 127(1):3–30.
- [Ting, 2000] Ting, K. M. (2000). A comparative study of cost-sensitive boosting algorithms. In *ICML*.
- [Topsøe, 1979] Topsøe, F. (1979). Information theoretical optimization techniques. *Kybernetika*, 15(1):8–27.
- [Tsochantaridis et al., 2004] Tsochantaridis, I., Hofmann, T., Joachims, T., and Altun, Y. (2004). Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the twenty-first international conference on Machine learning*, page 104. ACM.
- [Tsochantaridis et al., 2005] Tsochantaridis, I., Joachims, T., Hofmann, T., and Altun, Y. (2005). Large margin methods for structured and interdependent output variables. In *JMLR*, pages 1453–1484.
- [Turney, 1995] Turney, P. D. (1995). Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm. *Journal of artificial intelligence research*, pages 369–409.
- [von Neumann and Morgenstern, 1947] von Neumann, J. and Morgenstern, O. (1947). *Theory of Games and Economic Behavior*. Princeton University Press.
- [Wainwright and Jordan, 2008] Wainwright, M. J. and Jordan, M. I. (2008). Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305.
- [Wald, 1949] Wald, A. (1949). Statistical decision functions. *The Annals of Mathematical Statistics*, 20(2):165–205.
- [Wang and Tang, 2012] Wang, R. and Tang, K. (2012). Minimax classifier for uncertain costs. *arXiv preprint arXiv:1205.0406*.
- [Wolfowitz, 1950] Wolfowitz, J. (1950). Minimax estimates of the mean of a normal distribution with known variance. *The Annals of Mathematical Statistics*, pages 218–230.
- [Zadrozny et al., 2003] Zadrozny, B., Langford, J., and Abe, N. (2003). Cost-sensitive learning by cost-proportionate example weighting. In *ICDM*, pages 435–442.
- [Zhou and Liu, 2010] Zhou, Z.-H. and Liu, X.-Y. (2010). On multi-class cost-sensitive learning. *Computational Intelligence*, 26(3):232–257.

---

# Geometric Network Comparisons

---

**Dena Marie Asta**

Department of Engineering & Public Policy  
Department of Statistics  
Carnegie Mellon University  
Pittsburgh, PA 15213  
dasta@andrew.cmu.edu

**Cosma Rohilla Shalizi**

Department of Statistics  
Carnegie Mellon University  
Pittsburgh, PA 15213  
cshalizi@stat.cmu.edu

## Abstract

Network analysis needs tools to compare networks and assess the significance of differences between networks. We propose a principled statistical approach to network comparison that approximates networks as probability distributions on negatively curved manifolds. We outline the theory, as well as implement the approach on simulated networks, where its accuracy can be confirmed.

## 1 INTRODUCTION

Many scientific questions about networks amount to problems of *network comparison*: one wants to know whether networks observed at different times, or in different locations, or under different environmental or experimental conditions, actually differ in their structure. Such problems arise in neuroscience (e.g., comparing subjects with disease conditions to healthy controls, or the same subject before and after learning), in biology (e.g., comparing gene- or protein- interaction networks across species, developmental stages or cell types), and in social science (e.g., comparing different social relations within the same group, or comparing social groups which differ in some outcome). That the graphs being compared are not identical or even isomorphic is usually true, but scientifically unhelpful. What we need is a way to say if the difference between the graphs exceeds what we should expect from mere population variability or stochastic fluctuations. Network comparison, then, is a kind of two-sample testing, where we want to know whether the two samples could have come from the same source distribution. It is made challenging by the fact that the samples being compared are very structured, high-dimensional objects (networks), and more challenging because we often have only *one* graph in each sample.

We introduce a method for network comparison. The crucial idea is to approximate networks by continuous geo-

metric objects, namely probability densities, and then do two-sample bootstrap tests on those densities. Specifically, we draw on recent work showing how many real-world networks are naturally embedded in hyperbolic (negatively curved) manifolds. Graphs then correspond to clouds of points in hyperbolic space, and can be viewed as being generated by sampling from an underlying density on that space. We estimate a separate density for each of the two networks being compared, calculate the distance between those densities, and compare it to the distance expected under sampling from a pooled density estimate.

Our method, while conceptually fairly straightforward, is admittedly more complicated than the current practice in the scientific literature, which is to compare networks by taking differences in *ad hoc* descriptive statistics (e.g., average shortest path lengths, or degree distributions). It is very hard to assess the statistical significance of these differences, and counter-examples are known where the usual summary statistics fail to distinguish graphs which are qualitatively radically different (e.g., grid-like graphs from highly clustered tree-like ones). Similarly, whole-graph metrics and similarity measures are of little statistical use, without probability models to gauge their fluctuations. Below, we show through simulations that our method let us do network comparisons where (i) we can assess significance, (ii) power is high for qualitative differences, and (iii) when we detect differences, we also get some idea *how* the networks differ.

## 2 MOTIVATION AND BACKGROUND

A fundamental issue with network comparison, mentioned in the introduction, is that we often have *only* two networks to compare, and nonetheless need to make some assessment of statistical significance. This can obviously only be done by regarding the networks as being drawn from (one or more) probability models, and restricting the form of the model so that an observation of a single graph is informative about the underlying distribution. That is, we must restrict ourselves to network models which obey some sort



of law of large numbers or ergodic theorem within a single graph, or else we always have  $n = 1$ . As in any other testing problem, the better the alignment between the model's restrictions and actual properties of the graphs, the more efficiently the test will use the available information.

**Salient properties of actual networks** Over the last two decades, it has become clear that many networks encountered in the real world, whether natural or human-made, possess a number of mathematically striking properties (Newman, 2010). They have highly right-skewed degree distributions, they show the “small-world effect” of short average path lengths (growing only logarithmically with the number of nodes) but non-trivial transitivity of links, and high clusterability, often with a hierarchical arrangement of clusters. This is all a far cry from what is expected of conventional random graphs. While a large literature of parametric stochastic models has developed to try to account for these phenomena (Newman, 2010), there are few situations where a data analyst can *confidently* assert that one of these models is even approximately well-specified.

**Current approaches to network comparison** The typical approach in the literature is *ad hoc* comparison of common descriptive statistics on graphs (path lengths, clustering coefficients, etc.). These statistics are often misapplied, as in the numerous incorrect claims to have found “power law” or “scale-free” networks (Clauset et al., 2009), but that is not the fundamental issue. Even the recent authoritative review of, and advocacy for, the “connectomics” approach to neuroscience by Sporns (2010) takes this approach. Disturbingly, Henderson and Robinson (2011) show that, with commonly used choices of statistics and criteria, this approach cannot distinguish between complex, hierarchically-structured networks, and simple two-dimensional grids (such as a grid over the surface of the cortex).

More formally, Pao et al. (2011) study the power of tests based on such summaries to detect departures from the null hypothesis of completely independent and homogeneous edges (Erdos-Renyi graphs) in the direction of independent but heterogeneous edges. Their results were inconclusive, and neither their null nor the alternative models are plausible for real-world networks. Apart from this, essentially nothing is known about either the significance of such comparisons or their power, how to combine comparisons of different descriptive statistics, which statistics to use, or if significant differences are found, how to infer changes in structure from them. The issue of statistical significance also afflicts graph metrics and similarity measures, even those with plausible rationales in graph theory (e.g., that of Koutra et al. 2013).

Hunter et al. (2008) show one way to check goodness-of-fit for a model of a single network, using simulations to

check whether the observed values of various graph statistics are plausible under the model's sampling distribution. But they are unable to combine checks with different statistics, cannot find the power of such tests, and do not touch on differences across networks.

More relevantly to comparisons, Middendorf et al. (2005) use machine-learning techniques to classify networks as coming from one or another of various generative models, taking features of the network (such as the counts of small sub-graphs, or “motifs”) as the inputs to the classifier. They demonstrate good operating characteristics in simulations, but rely on having a good set of generative models to start with.

The approach to network comparison most similar to ours is Tang et al. (2014), which, like our proposed methods, models the nodes as drawn from densities on a latent space and attaches edges based on the geometric relationship between node coordinates. The primary difference between both approaches is the choice of latent space. Tang et al. (2014) use a Euclidean inner product space, allowing for an algebraic method of network inference. Our choice is motivated by the desire to pick a latent space that matches geometric properties of the real-world networks we aim to study.

A final related approach to network comparison is Rosvall and Bergstrom (2010), which like our proposed methods, uses bootstrap resampling from models fit to the original networks to assess significance of changes. The goal there however is not to detect global changes in the network structure, but local changes in which nodes are most closely tied to one another.

**Hyperbolic geometry of networks** While waiting for scientifically-grounded parametric models, we seek a class of non-parametric models which can accommodate the stylized facts of complex networks. Here we draw on the more recent observation that for many real-world networks, if we view them as metric spaces with distance given by shortest path lengths, the resulting geometry is *hyperbolic* (Albert et al., 2014, Kennedy et al., 2013, Krioukov et al., 2010), rather than Euclidean. Said another way, many real-world networks can be naturally embedded into negatively-curved continuous spaces. Indeed, Krioukov et al. (2010) show that if one draws points representing nodes according to a “quasi-uniform” distribution on the hyperbolic plane (see (2) below), and then connects nodes with a probability that decays according to the hyperbolic distance between the representative points, one naturally obtains graphs showing right-skewed degree distributions, short average path lengths, and high, hierarchical clusterability.

**Continuous latent space models** The model of (Krioukov et al., 2010) is an example of a *continuous latent*

space model, characterized by a metric space  $(M, \rho)$ , a link probability function  $W$ , and a probability density  $f$  on  $M$ , the *node density*. Points representing nodes are drawn iid from  $f$ , and edges form independently between nodes at  $x$  and  $y$  with probability  $W(x, y) = W(\rho(x, y))$  decreasing in the distance. As a hierarchical model,

$$\begin{aligned} Z_i &\sim_{iid} f \\ A_{ij}|Z_1, \dots, Z_n &\sim_{ind} W(\rho(Z_i, Z_j)) \end{aligned} \quad (1)$$

where  $A_{ij}$  is the indicator variable for an edge between nodes  $i$  and  $j$ . Holding  $M, \rho, W$  fixed, but allowing  $f$  to vary, we obtain different distributions over graphs. Two densities  $f, g$  on  $M$  determine the same distribution over graphs if  $f$  is the image of  $g$  under some isometry of  $(M, \rho)$ . Note that node *densities* can be compared regardless of the number of nodes in the observed graphs.

The best-known continuous latent space model for social networks is that of Hoff et al. (2002), where the metric space is taken to be Euclidean and the density  $f$  is assumed to be Gaussian. Our general methodology for network comparison could certainly be used with such models. However, the striking properties of large real-world graphs, such as their highly-skewed degree distributions, lead us to favor the sort of hyperbolic model used by Krioukov et al. (2010), but without their restrictive assumptions on  $f$ . Rather, we will show how to non-parametrically estimate the node density from a single observed graph, and then reduce network comparison to a comparison of these probability densities.

Continuous latent space models are themselves special cases of models called *graphons*, lifting the restriction that  $M$  be a metric space, and requiring of the edge probability function  $W(x, y)$  only that it be measurable and symmetric in its arguments<sup>1</sup>. Any distribution over infinite graphs which is invariant under permuting the order of the nodes turns out to be a mixture of such graphons (Kallenberg, 2005, ch. 7). Moreover, as one considers larger and larger graphs, the properties of the observed graph uniquely identify the generating graphon (Diaconis and Janson, 2008); what almost comes to the same thing, the limit of a sequence of growing graphs is a graphon (Borgs et al., 2006, Lovász, 2012, Borgs et al., 2014). One might, then, try to use our approach to compare graphons with estimated  $f$  and  $W$ . While graphon estimation is known to be possible in principle (Bickel et al., 2011, Choi and Wolfe, 2014), there are no published, computationally feasible methods to do it. Moreover, we expect to gain power by tailoring our models to enforce salient network properties, as described above. Accordingly, we turn to some of the important as-

<sup>1</sup>Graphons are often *defined* to have  $M = [0, 1]$  and  $f$  Lebesgue measure. One can show that any graphon over another measure space or with another node density is equivalent to one of this form, i.e., generates the same distribution over infinite graphs (Kallenberg, 2005, ch. 7).

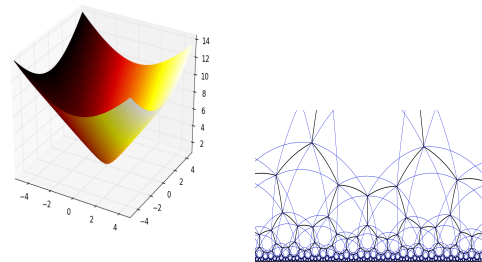


FIGURE 1: **Models of  $\mathbb{H}_2$**  A connected component of the hyperboloid  $x_3^2 = 1 + x_1^2 + x_2^2$  (left), with the metric given by the shortest possible Minkowski length of a path between points along the surface, is isometric to the Poincaré half-plane (right) under a suitable non-Euclidean metric. The half-plane is tiled into regions of equal area with respect to the metric. (Images from Rocchini (2007), under a Creative Commons license.)

pects of hyperbolic geometry.

## 2.1 HYPERBOLIC SPACES

Hyperbolic spaces are metric spaces which are negatively curved — the angles in a triangle of geodesics sum to less than 180 degrees. The oldest example of such a space is the surface of (one sheet of) the hyperboloid, the surface of points  $(x_1, x_2, x_3) \in \mathbb{R}^3$  such that

$$x_1^2 + x_2^2 - x_3^2 = -1,$$

with the distance between points taken to be the smallest possible Minkowski length of a path between them along the surface. Another, and perhaps even more basic, example of a hyperbolic space is a tree, again with the shortest-path metric. Our starting data will be observed networks, which are typically at least locally tree-like, and so also possess a hyperbolic geometry (Jonckheere et al., 2008).

As explained above, we aim to represent this discrete hyperbolic geometry with a density over a continuous hyperbolic space. For concreteness, we will focus on the hyperbolic plane  $\mathbb{H}_2$ , whose most basic geometric model is just the surface of the hyperboloid. It will be more convenient to work with another model of  $\mathbb{H}_2$ : the *Poincaré half-plane* of  $\mathbb{C}$ ,

$$\mathbb{H}_2 = \{x + iy \mid x \in \mathbb{R}, y \in (0, \infty)\}$$

equipped with the metric  $d\rho^2 = (dx^2 + dy^2)/y^2$ .

As mentioned above, (Krioukov et al., 2010) showed that if the density of nodes on the Poincaré half-plane is one of the *quasi-uniform* densities,

$$q_{\delta, R}(re^{i\theta}) = \frac{\delta \sinh \delta r}{2\pi(\sinh r) \cosh(\delta R - 1)}, \quad \delta > 0 \quad (2)$$

one obtains graphs which reproduce the stylized facts of right-skewed degree distributions, clusterability, etc., for

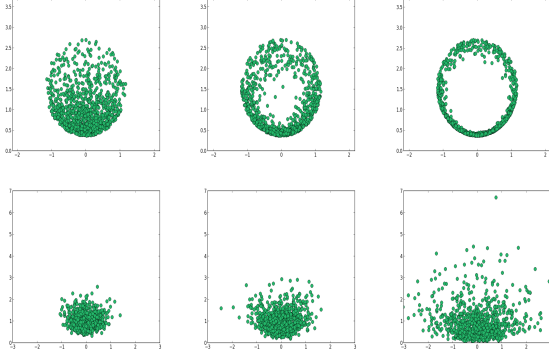


FIGURE 2: **Densities on  $\mathbb{H}_2$**  1000 points drawn iidly from quasi-uniform densities, Eq. 2 (top;  $\delta = 1, 10, 30$  from left to right,  $R = 1$  throughout), and from hyperbolic Gaussian densities, Eq. 8 (bottom,  $\sigma = 0.05, 0.1, 0.3$  from left to right).

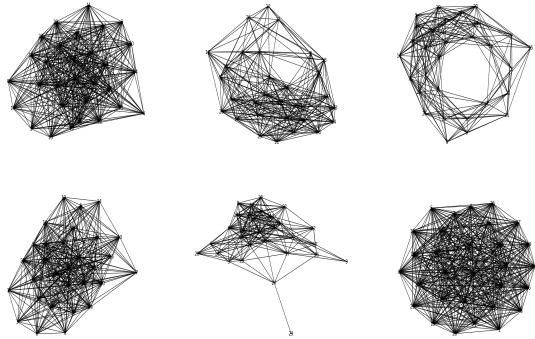


FIGURE 3: **Hyperbolic latent-space graphs** Graphs formed by drawing 30 node locations as in Fig. 2, and applying the link probability function  $W(x, y) = \Theta(\rho(x, y) - 1.5)$ . Note how the graphs in the bottom row become more clustered as the  $\delta$  parameter increases from left to right.

a wide range of link probability functions  $W$ , including Heaviside step functions  $\Theta(\rho - c)$ . Note that the mode of  $q$  is always at  $0 + i$ , with the parameter  $\delta > 0$  controlling the dispersion around the mode, and  $R > 0$  being an over-all scale factor. As  $\delta$  grows, the resulting graphs become more clustered.

We will introduce another family of densities on  $\mathbb{H}_2$ , the hyperbolic *Gaussians*, in the next section.

Fig. 2 shows samples from quasi-uniform distributions on  $\mathbb{H}_2$ , and Fig. 3 the resulting graphs. While we will use such networks as test cases, we emphasize that we will go beyond (2) to a fully nonparametric estimation of the node density.

### 3 METHOD

Our goal is to compare networks by comparing node densities. Our procedure for estimating node densities has in

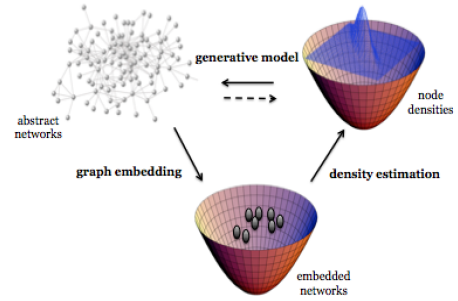


FIGURE 4: *Schematic of network inference*

turn two steps (Figure 4): we embed the nodes of an observed network into  $\mathbb{H}_2$  (§3.1), and then estimate a density from the embedded points (§3.2). We may then compare the observed difference between estimated node densities from two graphs to what would be expected if we observed two graphs drawn from a common node density (§3.3).

#### 3.1 GRAPH EMBEDDING

An *embedding* of a graph  $G$  is a mapping of its nodes  $V_G$  to points into a continuous metric space  $(M, \rho)$  which preserves the structure of the graph, or tries to. Specifically, the distances between the representative points should match the shortest-path distances between the nodes, as nearly as possible. This is a multidimensional scaling problem, where typically one seeks the embedding  $\phi : V_G \mapsto M$  minimizing

$$\sum_{(v,w) \in V_G^2} (\rho_G(v, w) - \rho(\phi(v), \phi(w)))^2, \quad (3)$$

where  $\rho_G$  is the shortest-path-length metric on  $V_G$ . Classically, when  $M = \mathbb{R}^n$  and  $\rho$  is the Euclidean metric, the arg-min of (3) can be found by spectral decomposition of the matrix of  $\rho_G(v, w)$  values (Hand et al., 2001, ch. 3).

Spectral decomposition does not however give the arg-min of (3) when  $M = \mathbb{H}_2$  with the appropriate non-Euclidean metric. While the solution could be approximated by gradient descent (Cvetkovski and Crovella, 2011), we follow Begelfor and Werman (2005) in changing the problem slightly. They propose minimizing

$$\sum_{(v,w) \in V_G^2} (\cosh \rho_G(v, w) - \cosh \rho(\phi(v), \phi(w)))^2 \quad (4)$$

which can be done exactly via a spectral decomposition. Specifically, let  $R_{ij} = \cosh \rho_G(i, j)$ , whose leading eigenvector is  $u_1$  and whose trailing eigenvectors are  $u_2$  and  $u_3$ . Then the  $i^{\text{th}}$  row of the matrix  $(u_1 u_2 u_3)$  gives the  $\mathbb{H}_2$  coordinates for node  $i$ . If  $R$  has one positive eigenvalue, exactly

2 negative eigenvalues, and all remaining eigenvalues vanish, this defines an exact isometric embedding (Begelfor and Werman, 2005).

We have not found a way of estimating the node density which avoids the initial step of embedding. Our method is, however, fairly indifferent as to *how* the nodes are embedded, so long as this is done well, and in a way which does not pre-judge the form of the node density.

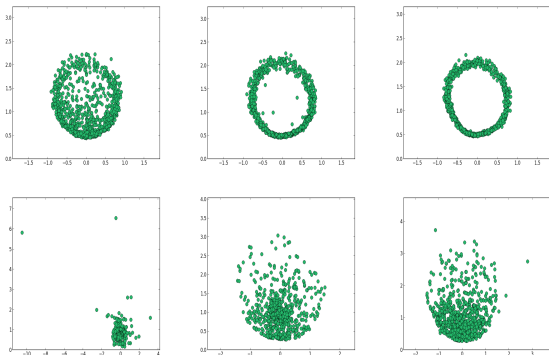


FIGURE 5: **Re-embedded Generated Graphs** Results of embedding simulated graphs, formed as in Fig. 3, back into  $\mathbb{H}_2$ . Comparison with Fig. 2 illustrates the fidelity of the embedding process.

### 3.2 DENSITY ESTIMATION

Having embedded the graph into  $\mathbb{H}_2$ , we estimate the node density. Our procedure for doing so is more easily grasped by first reviewing the connections between kernel density estimation, convolution, and Fourier transforms in Euclidean space.

**Kernel density estimation in Euclidean space as convolution** In Euclidean space, kernel density estimation smooths out the empirical distribution by adding a little bit of noise around each observation. Given observations  $z_1, z_2, \dots, z_n \in \mathbb{R}^p$ , and a normalized kernel function  $K_h$ , the ordinary kernel density estimator  $\hat{f}^{n,h}$  at a point  $z \in \mathbb{R}^p$  is

$$\begin{aligned} \hat{f}^{n,h}(z) &= \frac{1}{n} \sum_{i=1}^n K_h(z - z_i) \\ &= \int_{\mathbb{R}^p} K_h(z - z') \left( \frac{1}{n} \sum_{i=1}^n \delta(z' - z_i) \right) dz' \\ &= \int_{\mathbb{R}^p} K_h(z - z') \hat{P}_n(dz') \\ &= (K_h * \hat{P}_n)(z) \end{aligned}$$

where the third line defines the empirical measure  $\hat{P}_n$ , and  $*$  denotes convolution. In words, the kernel density estimate is the convolution of the empirical measure with the

kernel. Here the role of the kernel  $K_h$  is not so much to be a distribution over the Euclidean space, as a distribution over *translations* of the space:  $K_h(z - z_i)$  is really the density at the translation mapping the data point  $z_i$  into the operating point  $z$ . As it happens, the group of translations of  $\mathbb{R}^p$  is also  $\mathbb{R}^p$ , but when we adapt to non-Euclidean spaces, this simplifying coincidence goes away.

Since, in Euclidean space, the Fourier transform  $\mathcal{F}$  converts convolutions into products (Stein and Weiss, 1971),

$$\mathcal{F} \left[ \hat{f}^{n,h} \right] (s) = \mathcal{F} [K_h] (s) \mathcal{F} \left[ \hat{P}_n \right] (s)$$

This relation often greatly simplifies computing  $\hat{f}^{n,h}$ . It also lets us define the bandwidth  $h$ , through the relation  $\mathcal{F} [K_h] (s) = \mathcal{F} [K] (hs)$ .

It is well known that kernel density estimators on  $\mathbb{R}^p$ , with  $h \rightarrow 0$  at the appropriate rate in  $n$ , are minimax-optimal in their  $L_2$  risk (van der Vaart, 1998). With suitable modifications, this still holds for compact manifolds (Pelletier, 2005), but the hyperbolic plane  $\mathbb{H}_2$  is not compact.

#### 3.2.1 $\mathbb{H}_2$ -Kernel Density Estimator

Our method for density estimation on  $\mathbb{H}_2$  is a generalization of Euclidean kernel density estimation. In  $\mathbb{R}^p$ , the kernel is a density on translations of  $\mathbb{R}^p$ . For  $\mathbb{H}_2$ , the appropriate set of isometric transformations are not translations, but rather the class of “Möbius transformations” represented by the Lie group  $\mathbb{S}\mathbb{L}_2$  (Terras, 1985, Huckemann et al., 2010). An  $\mathbb{H}_2$  kernel, then, is a probability density on  $\mathbb{S}\mathbb{L}_2$ . We may write  $K_h(z, z_i)$  to abbreviate the density the kernel  $K_h$  assigns to the Möbius transform taking  $z_i$  to  $z$ . The generalized kernel density estimator on  $\mathbb{H}_2$  takes the form

$$\hat{f}^{n,h}(z) = \frac{1}{n} \sum_{i=1}^n K_h(z, z_i) \quad (5)$$

$$= (K_h * \hat{P}_n)(z) \quad (6)$$

In Euclidean space, the Fourier transform analyzes functions (or generalized functions, like  $\hat{P}_n$ ) into linear combinations of the eigenfunctions of the Laplacian operator. The corresponding operation for  $\mathbb{H}_2$  is the *Helgason*, or *Helgason-Fourier*, transform  $\mathcal{H}$  (Terras, 1985). The Fourier basis functions are indexed by  $\mathbb{R}^p$ , which is the group of translations; for analogous reasons, the Helgason basis functions are indexed by  $\mathbb{C} \times \mathbb{S}\mathbb{O}_2$ . Many of the formal properties of the Fourier transform carry over to the Helgason transform. (See App. A.) In particular, convolution still turns into multiplication:

$$\mathcal{H} \left[ \hat{f}^{n,h} \right] = \mathcal{H} [K_h] \mathcal{H} \left[ \hat{P}_n \right], \quad (7)$$

where  $\mathcal{H}[K_h]$  denotes the Helgason-Fourier transform of the well-defined density on  $\mathbb{H}_2$  induced by the density  $K_h$

on  $\mathbb{S}\mathbb{L}_2$ , and we define the bandwidth  $h$  through

$$\mathcal{H}[K_h](s, M) = \mathcal{H}[K](hs, M).$$

As in Euclidean density estimation,  $h$  may be set through cross-validation.

In a separate manuscript (Asta, 2014), we show that the  $L_2$  risk of (5) goes to zero at the minimax-optimal rate, under mild assumptions on the smoothness of the true density, and of the kernel  $K$ . (This is a special case of broader results about generalized kernel density estimation on symmetric spaces.) The assumptions on the kernel are satisfied by what Huckemann et al. (2010) calls “hyperbolic Gaussians”, densities on  $\mathbb{H}_2$  with parameter  $\rho$  defined through their Helgason transforms,

$$\mathcal{H}[K](s, M) \propto e^{\rho s(s-1)}. \quad (8)$$

Just as the ordinary Gaussian density is the unique solution to the heat equation with a point source in Euclidean space, the hyperbolic Gaussian is the unique ( $\mathbb{S}\mathbb{O}_2$ -invariant) solution to the heat equation on  $\mathbb{H}_2$  (Terras, 1985).

### 3.3 NETWORK COMPARISON

Combining embedding with kernel density estimation in  $\mathbb{H}_2$  gives us a method of estimating node densities, and so of estimating a hyperbolic latent space model for a given network. We now turn to *comparing* networks, by comparing these estimated node densities.

Our method follows the general strategy advocated in Genovese et al. (2013). Given two graphs  $G_1$  and  $G_2$ , we may estimate two separate network models

$$\hat{\mathcal{P}}_1 = \hat{\mathcal{P}}(G_1), \quad \hat{\mathcal{P}}_2 = \hat{\mathcal{P}}(G_2).$$

We may also pool the data from the two graphs to estimate a common model

$$\hat{\mathcal{P}}_{12} = \hat{\mathcal{P}}(G_1, G_2).$$

We calculate a distance  $d^* = d(\hat{\mathcal{P}}_1, \hat{\mathcal{P}}_2)$  using any suitable divergence. We then compare  $d^*$  to the distribution of distances which may be expected under the pooled model  $\hat{\mathcal{P}}_{12}$ . To do so, we independently generate  $G'_1, G'_2 \sim \hat{\mathcal{P}}_{12}$ , and calculate

$$d(\hat{\mathcal{P}}(G'_1), \hat{\mathcal{P}}(G'_2)).$$

That is, we bootstrap two independent graphs out of the pooled model, fit a model to each bootstrapped graph, and calculate the distance between them. Repeated over many bootstrap replicates, we obtain the sampling distribution of  $d$  under the null hypothesis that  $G_1$  and  $G_2$  are drawn from the same source, and any differences between them are due to population variability or stochastic fluctuations.<sup>2</sup>

<sup>2</sup>This method extends easily to comparing sets of graphs,  $G_{11}, G_{12}, \dots, G_{1n}$  vs.  $G_{21}, G_{22}, \dots, G_{2m}$ , but the notation grows cumbersome.

In our case, we have already explained how to find  $\hat{\mathcal{P}}_1$  and  $\hat{\mathcal{P}}_2$ . Since we hold the latent space  $M$  fixed at  $\mathbb{H}_2$ , and the link probability function  $W$  fixed, we can label our models by their node densities,  $\hat{f}_1^{n,h}$  and  $\hat{f}_2^{n,h}$ . To obtain the pooled model  $\hat{\mathcal{P}}_{12}$ , we first embed  $G_1$  and  $G_2$  separately using generalized multidimensional scaling, and then do kernel density estimation on the union of their embedded points.

The generalized multidimensional scaling technique we use depends only on the eigendecomposition of matrices determined by shortest path lengths. Therefore the  $L_2$  difference

$$\|\hat{f}_1^{n,h} - \hat{f}_2^{n,h}\|_2 \quad (9)$$

between two estimated node densities  $\hat{f}_1^{n,h}, \hat{f}_2^{n,h}$  is 0 if and only if the original sets of vertices from the different samples are isometric and hence (9) approximates a well-defined metric  $d$  on our continuous latent space models. Moreover, since the Plancherel identity carries over to the Helgason-Fourier transform (Terras, 1985),

$$d_2(f_1, f_2) = \|\mathcal{H}[f_1] - \mathcal{H}[f_2]\|_2, \quad (10)$$

and, for our estimated node densities,  $\mathcal{H}[f]$  is given by (7). Appendix B gives full details on our procedure for computing the test statistic (10).

### 3.4 THEORETICAL CONSIDERATIONS

Let us sum up our method, before turning to theoretical considerations. (0) We observe two graphs,  $G_1$  and  $G_2$ . (1) Through multi-dimensional scaling, we embed them separately in  $\mathbb{H}_2$  (§3.1), getting two point clouds, say  $\mathbf{Z}_1$  and  $\mathbf{Z}_2$ . (2) From each cloud, we estimate a probability density on  $\mathbb{H}_2$ , using hyperbolic Gaussian kernels, getting  $\hat{f}^{n_1, h_1}$  and  $\hat{f}^{n_2, h_2}$  (§3.2). We calculate  $\|\hat{f}^{n_1, h_1} - \hat{f}^{n_2, h_2}\|_2$  using (10). We also form a third density estimate,  $\hat{f}^{n_1+n_2, h_{12}}$ , from  $\mathbf{Z}_1 \cup \mathbf{Z}_2$ . (3) We generate two independent graphs  $G_1^*, G_2^*$  from  $\hat{f}^{n_1+n_2, h_{12}}$  according to (1), and subject these graphs to re-embedding and density estimation, obtaining  $\hat{f}^{n_1, h_1^*}$  and  $\hat{f}^{n_2, h_2^*}$  and so  $\|\hat{f}^{n_1, h_1^*} - \hat{f}^{n_2, h_2^*}\|_2$ . Finally, (4) repeating step (3) many times gives us the sampling distribution of the test statistic under the null hypothesis that  $G_1$  and  $G_2$  came from the same source, and the  $p$ -value is the quantile of  $\|\hat{f}^{n_1, h_1} - \hat{f}^{n_2, h_2}\|_2$  in this distribution.

The final step of computing the  $p$ -value is a fairly unproblematic bootstrap test. The previous step of generating new graphs from the pooled model is also an unproblematic example of a model-based bootstrap. The kernel density estimates themselves are consistent, and indeed converge at the minimax rate (Asta, 2014), *given* the point clouds on the hyperbolic plane. This makes it seem that the key step is the initial embedding. Certainly, it would be convenient if the graphs  $G_1$  and  $G_2$  were generated by a hyperbolic

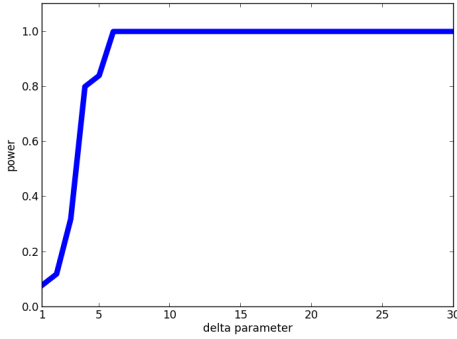


FIGURE 6: **Comparing Quasi-Uniforms** Power of our test, at size  $\alpha = 0.1$ , for detecting the difference between a 100-node graph generated from the quasi-uniform density  $q_{1,1}$  and a 100-node graph generated from  $q_{\delta,1}$ , as a function of the dispersion parameter  $\delta$ .

latent space model, and the embedding was a consistent estimator of the latent node locations. However, such strong conditions are not *necessary*. Suppose that if  $G_1 \sim \mathcal{P}_1$  and  $G_2 \sim \mathcal{P}_2 \neq \mathcal{P}_1$ , then  $\hat{f}_1^{n,h} \rightarrow f_1$  and  $\hat{f}_2^{n,h} \rightarrow f_2$ , with  $\|f_1 - f_2\|_2 > 0$ . Then at any nominal size (significance level)  $\alpha > 0$ , the power of the test will go to 1. For the nominal size of the test to match the actual size (probability of incorrectly rejecting the null hypothesis), however, will presumably require a closer alignment between the hyperbolic latent space model and the actual generating distribution.

## 4 SIMULATIONS

**Comparison of Graphs with Quasi-Uniform Node Densities** In our first set of simulation studies, we generated graphs which exactly conformed to the hyperbolic latent space model, and in fact ones where the node density was quasi-uniform (as in Fig. 3). One graph had 100 nodes, with latent locations drawn from a  $q_{1,1}$  distribution; the other, also of 100 nodes, followed a  $q_{\delta,1}$  distribution, with varying  $\delta$ . We used 50 bootstrap replicates (pairs of resampled networks) in each test, kept the nominal size  $\alpha = 0.1$ , and calculated power by averaging over 25 independent graph pairs (the number of power tests). Despite the graphs having only 100 nodes, Fig. 6 shows that our test has quite respectable power.

**Comparison of Watts-Strogatz Graphs** We have explained above, §2, why we expect hyperbolic latent space models to be reasonable ways of summarizing the structure of complex networks. However, they will also be more or less mis-specified for many networks of interest. We thus applied our methods to a class of graph distributions which do *not* follow a hyperbolic latent space model, namely Watts-Strogatz networks (Watts and Strogatz, 1998). Our

simulations used 100 node networks, with the base topology being a 1D ring with a branching factor of 40, and variable re-wiring probabilities. These graphs show the small-world property and high transitivity, but light-tailed degree distributions. Even in these cases, where the hyperbolic model is not the true generator, our comparison method had almost perfect power (Fig. 7).

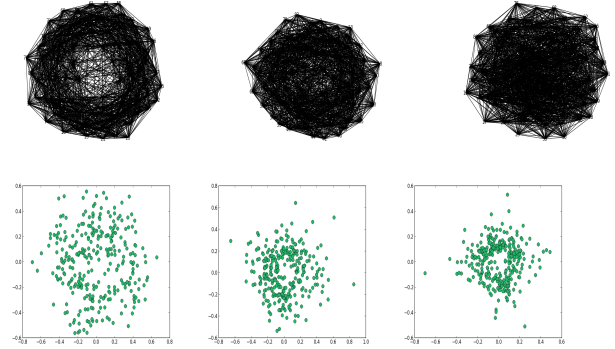


FIGURE 7: **Comparing Watts-Strogatz models** Above, Watts-Strogatz graphs formed by re-wiring 1D ring lattices (85 nodes, branching factor 40) with probability  $p$  per edge; from left to right  $p = 0.1, 0.2, 0.3$ . Below, embeddings of the graphs into  $\mathbb{H}_2$ . At nominal  $\alpha = 0.1$ , the power to detect these differences in  $p$  was 1.0 to within Monte Carlo error.

## 5 CONCLUSIONS

We have shown how nonparametric hyperbolic latent space models let us compare the global structures of networks. Our approach has its limits, and it may work poorly when the networks being compared are very far from hyperbolic. However, our experiments with Watts-Strogatz graphs show that it can detect differences among graph distributions from outside our model class. When we do detect a change in structure, we have a model for each network, namely their node densities, and the difference in node densities is an interpretable summary of how the networks differ. Many important directions for future work are now open. One important direction is a better handling of sparse networks, network growth, and the comparison of networks of different sizes — perhaps through some size-dependent modification of the link-probability function  $W$ , as in Krioukov et al. (2010), or the sort of scaling of graphons introduced in Borgs et al. (2014). But this should only extend our method’s scope.

## Acknowledgements

Our work was supported by NSF grant DMS-1418124, NIH grant R01 NS047493, and an NSF Graduate Research Fellowship under grant DGE-1252522. We are grateful for valuable discussions with Carl Bergstrom, Elizabeth Casman, David Choi, Aaron Clauset, Steve Fienberg,

Christopher Genovese, Dmitri Krioukov, Alessandro Rinaldo, Mitch Small, Andrew Thomas, Larry Wasserman, and Chris Wiggins, and for feedback from seminar audiences at UCLA's Institute for Pure and Applied Mathematics and CMU's machine learning and social science seminar.

## References

- Réka Albert, Bhaskar DasGupta, and Nasim Mobasher. Topological implications of negative curvature for biological and social networks. *Physical Review E*, 89: 032811, 2014. doi: 10.1103/PhysRevE.89.032811. URL <http://arxiv.org/abs/1403.1228>.
- Dena Asta. Kernel density estimation on symmetric spaces. arxiv:1411.4040, 2014. URL <http://arxiv.org/abs/1411.4040>.
- Evgeni Begelfor and Michael Werman. The world is not always flat, or, learning curved manifolds. Technical Report HUJI-CSE-LTR-2006-191, School of Engineering and Computer Science, Hebrew University of Jerusalem, 2005. URL <http://www.cs.huji.ac.il/~werman/Papers/cmds.pdf>.
- Peter J. Bickel, Aiyu Chen, and Elizaveta Levina. The method of moments and degree distributions for network models. *Annals of Statistics*, 39:38–59, 2011. URL <http://arxiv.org/abs/1202.5101>.
- Christian Borgs, Jennifer T. Chayes, László Lovász, Vera T. Sós, Balázs Szegedy, and Katalin Vesztegombi. Graph limits and parameter testing. In *Proceedings of the 38th Annual ACM Symposium on the Theory of Computing [STOC 2006]*, pages 261–270, New York, 2006. ACM. URL <http://research.microsoft.com/en-us/um/people/jchayes/Papers/TestStoc.pdf>.
- Christian Borgs, Jennifer T. Chayes, Henry Cohn, and Yufei Zhao. An  $L^p$  theory of sparse graph convergence I: Limits, sparse random graph models, and power law distributions. arxiv:1401.2906, 2014. URL <http://arxiv.org/abs/1401.2906>.
- David S. Choi and Patrick J. Wolfe. Co-clustering separately exchangeable network data. *Annals of Statistics*, 42:29–63, 2014. doi: 10.1214/13-AOS1173. URL <http://arxiv.org/abs/1212.4093>.
- Aaron Clauset, Cosma Rohilla Shalizi, and M. E. J. Newman. Power-law distributions in empirical data. *SIAM Review*, 51:661–703, 2009. URL <http://arxiv.org/abs/0706.1062>.
- Andrej Cvetkovski and Mark Crovella. Multidimensional scaling in the Poincaré disk. E-print, 2011. URL <http://arxiv.org/abs/1105.5332>.
- Persi Diaconis and Svante Janson. Graph limits and exchangeable random graphs. *Rendiconti di Matematica e delle sue Applicazioni*, 28:33–61, 2008. URL <http://arxiv.org/abs/0712.2749>.
- Christopher Genovese, Cosma Rohilla Shalizi, and Andrew C. Thomas. Network comparisons. Manuscript in preparation, 2013.
- David Hand, Heikki Mannila, and Padhraic Smyth. *Principles of Data Mining*. MIT Press, Cambridge, Massachusetts, 2001.
- J. A. Henderson and P. A. Robinson. Geometric effects on complex network structure in the cortex. *Physical Review Letters*, 107:018102, 2011. doi: 10.1103/PhysRevLett.107.018102.
- Peter D. Hoff, Adrian E. Raftery, and Mark S. Handcock. Latent space approaches to social network analysis. *Journal of the American Statistical Association*, 97:1090–1098, 2002. URL <http://www.stat.washington.edu/research/reports/2001/tr399.pdf>.
- Stephan F. Huckemann, Peter T. Kim, Ja-Yong Koo, and Axel Munk. Möbius deconvolution on the hyperbolic plane with application to impedance density estimation. *Annals of Statistics*, 38:2465–2498, 2010. doi: 10.1214/09-AOS783. URL <http://arxiv.org/abs/1010.4202>.
- David R. Hunter, Steven M. Goodreau, and Mark S. Handcock. Goodness of fit of social network models. *Journal of the American Statistical Association*, 103:248–258, 2008. doi: 10.1198/016214507000000446. URL <http://www.csss.washington.edu/Papers/wp47.pdf>.
- Edmond Jonckheere, Poonsuk Lohsoonthorn, and Francis Bonahon. Scaled Gromov hyperbolic graphs. *Journal of Graph Theory*, pages 157–180, 2008. doi: 10.1002/jgt.20275. URL [http://eudoxus2.usc.edu/jgt6396\\_final.pdf](http://eudoxus2.usc.edu/jgt6396_final.pdf).
- Olav Kallenberg. *Probabilistic Symmetries and Invariance Principles*. Springer-Verlag, New York, 2005.
- W. Sean Kennedy, Onuttom Narayan, and Iraj Saniee. On the hyperbolicity of large-scale networks. arxiv:1307.0031, 2013. URL <http://arxiv.org/abs/1307.0031>.
- Danai Koutra, Joshua T. Vogelstein, and Christos Faloutsos. DELTACON: A principled massive-graph similarity function. In Joydeep Ghosh, Zoran Obradovic, Jennifer Dy, Zhi-Hua Zhou, Chandrika Kamath, and Srinivasan Parthasarathy, editors, *SIAM International Conference in Data Mining [SDM 2013]*, pages 162–170, Philadelphia, 2013. Society for Industrial and Applied Mathematics. doi: 10.1137/1.9781611972832.18. URL <http://arxiv.org/abs/1304.4657>.
- Dmitri Krioukov, Fragkiskos Papadopoulos, Maksim Kitsak, Amin Vahdat, and Marián Boguñá. Hyperbolic geometry of complex networks. *Physical Review E*, 82:

- 036106, 2010. doi: 10.1103/PhysRevE.82.036106. URL <http://arxiv.org/abs/1006.5169>.
- László Lovász. *Large Networks and Graph Limits*. American Mathematical Society, Providence, Rhode Island, 2012.
- Manuel Middendorf, Etay Ziv, and Chris Wiggins. Inferring network mechanisms: The *drosophila melanogaster* protein interaction network. *Proceedings of the National Academy of Sciences (USA)*, 102:3192–3197, 2005. URL <http://arxiv.org/abs/q-bio/0408010>.
- Mark E. J. Newman. *Networks: An Introduction*. Oxford University Press, Oxford, England, 2010.
- Henry Pao, Glen A. Coppersmith, and Carey E. Priebe. Statistical inference on random graphs: Comparative power analyses via Monte Carlo. *Journal of Computational and Graphical Statistics*, 20:395–416, 2011. doi: 10.1198/jcgs.2010.09004.
- Bruno Pelletier. Kernel density estimation on Riemannian manifolds. *Statistics and Probability Letters*, 73:297–304, 2005. doi: 10.1016/j.spl.2005.04.004.
- Claudio Rocchini. Poincare halfplane eptagonal hb, 2007. URL [http://commons.wikimedia.org/wiki/File:Poincare\\_halfplane\\_eptagonal\\_hb.svg](http://commons.wikimedia.org/wiki/File:Poincare_halfplane_eptagonal_hb.svg). Retrieved 20 October 2014.
- Martin Rosvall and Carl T. Bergstrom. Mapping change in large networks. *PLoS ONE*, 5:e8694, 2010. doi: 10.1371/journal.pone.0008694. URL <http://arxiv.org/abs/0812.1242>.
- Olaf Sporns. *Networks of the Brain*. MIT Press, Cambridge, Massachusetts, 2010.
- Elias M Stein and Guido L Weiss. *Introduction to Fourier analysis on Euclidean spaces*, volume 1. Princeton university press, 1971.
- Minh Tang, Avanti Athreya, Daniel L. Sussman, Vince Lyzinski, and Carey E. Priebe. A nonparametric two-sample hypothesis testing problem for random dot product graphs. E-print, arxiv.org, 2014. URL <http://arxiv.org/abs/1409.2344>.
- Audrey Terras. *Harmonic analysis on symmetric spaces and applications*, volume 1. Springer-Verlag, New York, 1985.
- A. W. van der Vaart. *Asymptotic Statistics*. Cambridge University Press, Cambridge, England, 1998.
- Duncan J. Watts and Steven H. Strogatz. Collective dynamics of “small-world” networks. *Nature*, 393:440–442, 1998. doi: 10.1038/30918.



---

# Learning and Planning with Timing Information in Markov Decision Processes

---

Pierre-Luc Bacon, Borja Balle, and Doina Precup  
Reasoning and Learning Lab,  
School of Computer Science, McGill University  
{pbacon,bballe,dprecup}@cs.mcgill.ca

## Abstract

We consider the problem of learning and planning in Markov decision processes with temporally extended actions represented in the options framework. We propose to use predictions about the duration of extended actions to represent the state and show that this leads to a compact predictive state representation model independent of the set of primitive actions. Then we develop a consistent and efficient spectral learning algorithm for such models. Using just the timing information to represent states allows for faster improvement in the planning performance. We illustrate our approach with experiments in both synthetic and robot navigation domains.

## 1 INTRODUCTION

Modelling the dynamics of an agent embedded in a large, complex environment is key to building good planning algorithms for such agents. In most practical applications, models are carefully designed by hand, and the agent’s “state” is given by measurements which are understandable by the designer of the system (such as spatial location and velocity, in the case of a robot). However, learning dynamical models for such states from data, as well as planning with them can be quite tricky. An alternative idea is to use models that are “subjective”, centered on the agent’s own perception and action capabilities. For example, affordances [Gibson, 1977] describe “state” through the courses of action that are enabled. Similarly, in robotics, subjective representations have been used to model dynamics, e.g. [Bowling *et al.*, 2005; Stober *et al.*, 2011]. Such models are appealing from a psychological point of view, but run into computational problems in very large observation spaces.

In this paper, we focus on a special class of subjective models, *timing models*, which arise from restricting the observations available to the agent to just information about the

duration of certain courses of action. Timing of events is understood to be crucial to animal learning [Machado *et al.*, 2009]. The goal of this paper, however, is not learning of the timing of external events, but rather to learn the duration of courses of action that an agent might take. The ensemble of such durations will constitute the agent’s *state*, which will be maintained as new data is received. We use the framework of options [Sutton *et al.*, 1999] to model extended courses of actions, and we present an approach for learning a predictive model of option durations.

Our models over durations can be viewed as affordances if we consider an option to be available when its predicted duration is within some reasonable bounds. Note that these models are much simpler than full option models, which provide joint information on the timing as well as the state or observation in which the option will terminate, e.g. [Wolfe and Singh, 2006]. We emphasize that timing information is very cheap to measure and process, even with simple hardware. Hence, a major area of application for this type of approach is on devices in which processing can be costly, such as simple, Roomba-like robots (as in our experiments) or cellphones (on which using a lot of sensors or computation drains the battery).

Our approach can also be interpreted as a computationally and statistically efficient way of exploiting prior information about useful courses of action provided in the form of options. The size of our models is independent of the number of possible primitive actions in the underlying system, which is very useful in large or continuous action spaces. Moreover, because we are able to learn feature representations for states using timing information only, our method can be applied to observable settings with high-dimensional, complex observations, as well as to partially observable settings, where building a full model for planning would be too data and computation-hungry. Examples include investment problems (where the best strategy may include information about past performance, past trades, news etc, which is too hard to process) or robotics (where sensors may produce a lot of data, but this may not be easy to process in real-time).

Of course, the utility of timing models depends on the nature of the task to be solved by the agent, as well as on the “quality” of the options available to the agent. The simplest example in which option duration models are beneficial is that of minimum time to goal problems, in which an agent receives a fixed penalty per time step until its task is completed. In this case, knowing the duration of an option immediately provides the reward model, so the option duration model has direct value for a planner. More generally, option duration models are beneficial as a form of localization. If you imagine a robot that has models for options that move radially out from the current position, this would allow localizing with respect to all neighboring walls. Finally, consider a problem in which a financial agent is holding stocks, and options which hold a particular stock while it is above a certain value, and sell under that value. In this case, timing models tell us exactly when stocks would be crossing certain barriers. It is clear in this case that, even though we are estimating only durations, the model encodes important state information (because of the way in which the options are defined).

In this paper, we analyze the capacity of option duration models to represent states in a Markov Decision Process (MDP). We propose a spectral algorithm for learning option duration models which builds on existing work for learning transformed predictive state representations [Rosencrantz *et al.*, 2004]. Finally we evaluate the quality of learning and planning with this type of models in experiments with discrete and continuous MDPs.

## 2 PRELIMINARIES AND NOTATION

We use bold letters to denote vectors  $\mathbf{v} \in \mathbb{R}^d$  and matrices  $\mathbf{M} \in \mathbb{R}^{d_1 \times d_2}$ . For matrix  $\mathbf{M}$ ,  $\mathbf{M}^+$  denotes its Moore–Penrose pseudo-inverse. Sometimes we name the columns and rows of a matrix using ordered index sets  $\mathcal{I}$  and  $\mathcal{J}$ , so  $\mathbf{M} \in \mathbb{R}^{\mathcal{I} \times \mathcal{J}}$  denotes a matrix of size  $|\mathcal{I}| \times |\mathcal{J}|$  with rows and columns indexed by  $\mathcal{I}$  and  $\mathcal{J}$  respectively.

Let  $\Sigma$  be a finite set of symbols. We use  $\Sigma^*$  to denote the set of all finite strings over  $\Sigma$  and  $\lambda$  for the empty string. Given two strings  $u, v \in \Sigma^*$ ,  $w = uv$  is their concatenation, in which case  $u$  is called a prefix of  $w$ , and  $v$  is a suffix of  $w$ . Given two sets of strings  $\mathcal{P}, \mathcal{S} \subseteq \Sigma^*$ ,  $\mathcal{PS}$  is the set obtained by taking every string of the form  $uv$  with  $u \in \mathcal{P}$  and  $v \in \mathcal{S}$ .

Given a predicate  $\mu(x)$  we denote by  $\mathbb{I}[\mu(x)]$  the indicator function which is one when  $\mu(x)$  is true and zero otherwise.

### 2.1 Markov Decision Processes and Temporally Extended Actions

A *Markov decision process* (MDP) is a tuple  $M = \langle S, A, P, R \rangle$  where  $S$  is the state set,  $A$  is the action set,  $P : S \times A \rightarrow (S \rightarrow [0, 1])$  defines a probability distribu-

tion over next states, and  $R : S \times A \rightarrow \mathbb{R}$  is the expected reward function (see [Puterman, 1994] for a review). We refer to probability distributions on  $S$  by  $\alpha$ , but sometimes use  $\boldsymbol{\alpha}$  to stress that we view them as vectors in  $\mathbb{R}^S$ . Suppose  $\alpha$  is a distribution over  $S$  and  $\pi : S \times A \rightarrow [0, 1]$  is a stochastic action policy which, given state  $s$ , chooses action  $a$  with probability  $\pi(s, a)$ . The environment then returns a state sampled from  $P$ ; and the resulting state distribution  $\alpha'$  is given by:

$$\alpha'(s') = \sum_{s \in S} \alpha(s) \sum_{a \in A} \pi(s, a) P(s, a)(s') .$$

Temporal abstraction in MDPs has been used as a tool to speed up learning and planning algorithms. We adopt the framework of options [Sutton *et al.*, 1999], with the goal of learning state representations based on option timing models. An *option* is a tuple  $\omega = \langle I_\omega, \pi_\omega, \beta_\omega \rangle$  where  $I_\omega \subseteq S$  is the set of initial states,  $\pi_\omega : S \times A \rightarrow [0, 1]$  is the option’s stochastic action policy, and  $\beta_\omega : S \rightarrow [0, 1]$  is the option termination probability for each state. We will drop the option’s subscript and write  $\omega = \langle I, \pi, \beta \rangle$  when there is no risk of confusion.

We say that an agent *interacts with an MDP via a set of options*  $\Omega$  if at all times the actions performed by the agent follow the policy specified by some option  $\omega \in \Omega$ , which is executed until termination.

Each option has an associated reward model  $R(s, \omega)$  and an associated transition model  $P(s, \omega)(s')$ , which can be computed using the MDP model and the definition of the options (see [Sutton *et al.*, 1999] for details). Given a set of options  $\Omega$ , the optimal option-value function satisfies the following Bellman equation:

$$Q_\Omega^*(s, \omega) = R(s, \omega) + \sum_{s' \in S} P(s, \omega)(s') \max_{\omega' \in \Omega} Q_\Omega^*(s', \omega')$$

Planning with options aims to find the optimal policy over options  $\pi_\Omega^*$ , which is achieved most often by estimating  $Q_\Omega^*$ . This does not require a model to be known; instead, it can be done using samples, in similar fashion as Q-learning. If the state space is large or continuous, function approximation can be used to represent states. The option duration model that we develop in this paper will be used as a form of state representation in order to learn an approximation to  $Q_\Omega^*$ .

### 2.2 Predictive State Representations

A *predictive state representation* is a model of a dynamical system in which the current state is represented as a set of predictions about the future behavior of the system [Littman *et al.*, 2002; Singh *et al.*, 2004]. We use a particular instantiation of this general idea, the so-called *transformed linear predictive state representation* [Rosencrantz *et al.*, 2004], abbreviated as PSR.

Let  $\Sigma$  be a finite set of symbols. A PSR over  $\Sigma$  is a tuple  $\mathcal{A} = \langle \alpha_\lambda, \alpha_\infty, \{\mathbf{A}_\sigma\}_{\sigma \in \Sigma} \rangle$  where  $\alpha_\lambda, \alpha_\infty \in \mathbb{R}^n$  are the initial and final weights respectively, and  $\mathbf{A}_\sigma \in \mathbb{R}^{n \times n}$  are the transition weights. The dimension  $n$  of these vectors and matrices is the *number of states* of the PSR. Though PSR is the usual name for this type of model in the reinforcement learning literature, they are also called *weighted finite automaton* (WFA) [Droste and Vogler, 2009] or *observable operator models* (OOM) [Jaeger, 2000]. This distinction reflects the fact that this same parametrization can be used to define models with different semantics, depending on the meaning associated with the values computed by the model.

A PSR  $\mathcal{A}$  computes a function  $f_{\mathcal{A}} : \Sigma^* \rightarrow \mathbb{R}$  that assigns a number to each string  $x = x_1 x_2 \cdots x_t \in \Sigma^*$  as follows:

$$f_{\mathcal{A}}(x) = \alpha_\lambda^\top \mathbf{A}_{x_1} \mathbf{A}_{x_2} \cdots \mathbf{A}_{x_t} \alpha_\infty = \alpha_\lambda^\top \mathbf{A}_x \alpha_\infty .$$

In a PSR, a string  $x \in \Sigma^*$  represents a sequence of events produced by a dynamical system, and  $f_{\mathcal{A}}(x)$  is the probability that the system produces  $x$ . In many cases of interest,  $\Sigma = A \times O$ , where  $A$  is a set of actions and  $O$  a set of observations, so any  $x \in \Sigma^*$  represents a sequence of action-observation pairs. The vector  $\alpha_\lambda$  represents the initial state of the system.

If a history  $u \in \Sigma^*$  has already been observed, the conditional probability of observing a sequence of events  $v \in \Sigma^*$  after  $u$  is:

$$f_{\mathcal{A},u}(v) = \frac{f_{\mathcal{A}}(uv)}{f_{\mathcal{A}}(u)} = \frac{\alpha_\lambda^\top \mathbf{A}_u \mathbf{A}_v \alpha_\infty}{\alpha_\lambda^\top \mathbf{A}_u \alpha_\infty} = \frac{\alpha_u^\top \mathbf{A}_v \alpha_\infty}{\alpha_u^\top \alpha_\infty} .$$

Hence, given some history  $u$ , the initial state  $\alpha_\lambda$  can be updated to a new state  $\alpha_u / (\alpha_u^\top \alpha_\infty)$ , which allows computing probabilities of future observations conditioned on the current history. Because the partition of a sequence of observations  $x$  into a history  $u$  and a future  $v$  (also called *test*) yields  $x = uv$ , we sometimes call histories *prefixes* and futures *suffixes*.

### 2.3 Hankel and System Dynamics Matrices

The behavior of a stochastic dynamical system producing observations in a finite set  $\Sigma$  can be entirely characterized by the function  $f : \Sigma^* \rightarrow \mathbb{R}$  giving the probability  $f(x)$  of observing each possible sequence of observations  $x$ . A convenient algebraic way to summarize all the information conveyed by  $f$  is with its *Hankel matrix*, a bi-infinite matrix  $\mathbf{H}_f \in \mathbb{R}^{\Sigma^* \times \Sigma^*}$  with rows and columns indexed by strings in  $\Sigma^*$ . Strings indexing rows and columns are interpreted as prefixes and suffixes respectively. The entries in  $\mathbf{H}_f$  are given by  $\mathbf{H}_f(u, v) = f(u, v)$  for every  $u, v \in \Sigma^*$ .

Although  $\mathbf{H}_f$  is an infinite matrix, in some cases it can have finite rank. In particular, a well-known result states that  $\mathbf{H}_f$  has rank at most  $n$  if and only if there exists a PSR  $\mathcal{A}$

with  $n$  states satisfying  $f_{\mathcal{A}} = f$  [Carlyle and Paz, 1971; Fliess, 1974]. This result is the basis of recently developed spectral learning algorithms for PSRs [Boots *et al.*, 2011], which we review in Sec. 4.

Instead of looking at the full Hankel matrix, algorithms usually work with finite sub-blocks of this matrix. A convenient way to specify such blocks is to give the “names” to the rows and columns. Specifically, given a finite set of prefixes  $\mathcal{P} \subset \Sigma^*$  and a finite set of suffixes  $\mathcal{S} \subset \Sigma^*$ , the pair  $\mathcal{B} = (\mathcal{P}, \mathcal{S})$  is a *basis* defining the sub-block  $\mathbf{H}_{\mathcal{B}} \in \mathbb{R}^{\mathcal{P} \times \mathcal{S}}$  of  $\mathbf{H}_f$ , whose entries are given by  $\mathbf{H}_{\mathcal{B}}(u, v) = \mathbf{H}_f(u, v)$ . Note that every sub-block built in this way satisfies  $\text{rank}(\mathbf{H}_{\mathcal{B}}) \leq \text{rank}(\mathbf{H}_f)$ ; when equality is attained, the basis  $\mathcal{B}$  is *complete*.

Sometimes it is also convenient to look at one-step shifts of the finite Hankel matrices. Let  $\mathbf{H} \in \mathbb{R}^{\mathcal{P} \times \mathcal{S}}$  be a finite sub-block of  $\mathbf{H}_f$  specified by a basis  $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ . Then, for every symbol  $\sigma \in \Sigma$ , we define the sub-block  $\mathbf{H}_\sigma \in \mathbb{R}^{\mathcal{P} \times \mathcal{S}}$  whose entries are given by  $\mathbf{H}_\sigma(u, v) = \mathbf{H}_f(u, \sigma v)$ . For a fixed basis, we also consider the vectors  $\mathbf{h}_{\mathcal{S}} \in \mathbb{R}^{\mathcal{S}}$  with entries given by  $\mathbf{h}_{\mathcal{S}}(v) = \mathbf{H}_f(\lambda, v)$  for every  $v \in \mathcal{S}$ , and  $\mathbf{h}_{\mathcal{P}} \in \mathbb{R}^{\mathcal{P}}$  with  $\mathbf{h}_{\mathcal{P}}(u) = \mathbf{H}_f(u, \lambda)$ .

The Hankel matrix  $\mathbf{H}_f$  is tightly related to the *system dynamics matrix* (SDM) of the stochastic process described by  $f$  [Singh *et al.*, 2004], but while the entries of the Hankel matrix represent *joint* probabilities over prefixes and suffixes, the corresponding entry in the SDM is the *conditional* probability of observing a suffix given the prefix. In particular, the SDM of  $f$  is the matrix  $\tilde{\mathbf{H}}_f \in \mathbb{R}^{\Sigma^* \times \Sigma^*}$  given by  $\mathbf{H}_f$  with the rows scaled by the probability of prefixes. The SDM of  $f$  shares a lot of properties with its Hankel counterpart. In particular, spectral learning algorithms for  $f$  can be derived in terms of sub-blocks of both  $\mathbf{H}_f$  and  $\tilde{\mathbf{H}}_f$ . We will work with the Hankel matrices of  $f$ , but all our algorithms can easily be adapted to work with the SDM instead.

## 3 OPTION DURATION MODELS

We are interested in the dynamics of an agent interacting with an MDP  $M$  via a set of options  $\Omega$ . Recall that in this setting the agent is not allowed to perform primitive actions, and options must be executed until termination. We are interested in situations in which the *duration* of an option is an informative statistic about the state of the MDP. That is, we would like to identify a state in  $M$  with the distribution over the durations of a sequence of options executed starting from that state. This section introduces a special PSR for this purpose that we call the *option duration model* (ODM). This is basically a dynamical model of an MDP viewed through the lens of a fixed set of options, in which the state is represented by a vector of parameters sufficient for predicting the duration of future options. In the following sections we present a spectral algorithm for

learning ODM from just information about the duration of options in an MDP, and explore the possibilities the state representation given by this ODM in order to plan with options in the original MDP.

The goal of this section is to understand what form the probability distributions over option durations take, and how to write these compactly in order to allow efficient learning from just duration information. To begin, assume we can reset the MDP  $M$  to a fixed state  $s_0$  and explore the environment from there by performing a sequence of  $t$  options  $\omega_1, \dots, \omega_t$ , e.g. chosen uniformly at random. From the point of view of the MDP dynamics, this process will generate a sequence of of state-action trajectories like the following:

$$\begin{aligned} \omega_1 &: (s_0, \pi_{\omega_1}(s_0), s_1, \pi_{\omega_1}(s_1), \dots, s_{T_1-1}, \pi_{\omega_1}(s_{T_1-1}), s_{T_1}) \\ \omega_2 &: (s_{T_1}, \pi_{\omega_2}(s_{T_1}), s_{T_1+1}, \pi_{\omega_2}(s_{T_1+1}), \dots, \pi_{\omega_2}(s_{T_2-1}), s_{T_2}) \\ &\dots \\ \omega_t &: (s_{T_{t-1}}, \pi_{\omega_t}(s_{T_{t-1}}), \dots, \pi_{\omega_t}(s_{T_t-1}), s_{T_t}) \end{aligned}$$

where  $\pi_{\omega}(s)$  denotes an action chosen according to the distribution given by  $\omega$ 's stochastic policy. Note that in this trace, the sequence of visited states, the actions executed, and the option stopping times form a stochastic dynamical process whose distribution can be computed in terms of the parameters of the MDP  $M$ , the parameters defining the options in  $\Omega$ , and the sequence of options  $\omega_1, \dots, \omega_t$ .

If one is only interested in the duration of the options executed in the above trace, then the interaction can be summarized in terms of termination and continuation events for the options being executed. That is, at each time step, we can ignore the current observed state and the action being chosen, and just focus on whether the current option terminates in the current state. We will use the symbol  $\sharp$  (*sharp*) to denote continuation, and  $\perp$  (*bottom*) to denote termination. Thus, at a given time step, we use  $(\omega, \sharp)$  to denote option  $\omega$  is being executed and does not terminate, and  $(\omega, \perp)$  if it terminates. Using this notation, the ‘‘duration information’’ in the previous trajectory can be expressed as:

$$(\omega_1, \sharp)^{d_1-1}(\omega_1, \perp)(\omega_2, \sharp)^{d_2-1}(\omega_2, \perp) \dots (\omega_t, \sharp)^{d_t-1}(\omega_t, \perp) \text{ ,}$$

where the durations are given by  $d_i = T_i - T_{i-1}$ ,  $T_0 = 0$ . In more compact form, we can write the duration information in a trace as a sequence of option-duration pairs:  $(\omega_1, d_1)(\omega_2, d_2) \dots (\omega_t, d_t)$ , with  $\omega_i \in \Omega$  and  $d_i \in \mathbb{N} = \{1, 2, \dots\}$ . Both notations will be convenient in our derivations.

Broadly speaking, an option duration model for an MDP  $M$  and a set of options  $\Omega$  is a mapping that associates with every state  $s$  in  $M$  a conditional probability distribution

$$\mathbb{P}_s[\vec{d} \mid \bar{\omega}] = \mathbb{P}_s[d_1, \dots, d_t \mid \omega_1, \dots, \omega_t]$$

that represents the stochastic process of stopping events induced by executing the options  $\bar{\omega} = \omega_1, \dots, \omega_t$  in  $M$  starting from  $s$ . More formally, let  $\mathcal{D}_t$  denote the set of

all<sup>1</sup> probability distributions over  $\mathbb{N}^t$ , and for every state  $s$  and  $\bar{\omega} \in \Omega^t$  let  $D_{\bar{\omega}}^s \in \mathcal{D}_t$  denote the distribution  $\mathbb{P}_s[\cdot \mid \bar{\omega}]$ . Now, by letting  $\bar{\omega}$  run over all possible sequence of options in  $\Omega^+$  we can associate with every  $s$  a mapping  $D_{\bullet}^s : \Omega^+ \rightarrow \cup_{t>0} \mathcal{D}_t$ . We say that the set of options  $\Omega$  is *rich* for MDP  $M$  if the maps  $D_{\bullet}^s$  are different for all  $s$ , that is, if the functional  $\Delta(s) = D_{\bullet}^s$  is injective, so for every  $s, s' \in S$  there exists some sequence of options  $\bar{\omega} \in \Omega^+$  such that  $D_{\bar{\omega}}^s \neq D_{\bar{\omega}}^{s'}$ . Clearly, in this case,  $\Delta(s)$  uniquely identifies the state  $s$ , indicating that the duration over options provides a unique representation for each state.

If  $\Delta$  is not injective, such models can still be sufficient for planning in special circumstances. For example, consider minimum-time-to-goal problems, in which fixed negative rewards are attributed per time step, and suppose the agent intends to plan using only options. In this case, states for which  $\Delta(s) = \Delta(s')$  will also have the same optimal value function  $V_{\Omega}^*$  (a result that follows directly from the way in which option models are defined [Sutton *et al.*, 1999]).

In next section we give a concrete realization of this abstract ODM by showing how to encode the collection of distributions  $\Delta(s)$  using a PSR.

### 3.1 Representing Option Duration Models with PSR

We now show that the probability distributions of the timing of options can be compactly represented in the form of a PSR. Given  $s \in S$  and an option  $\omega \in \Omega$ , we denote by  $\delta(s, \omega)$  the random variable counting the number of steps until termination when following  $\omega$  from  $s$ . Note that  $s$  might be an initial state for  $\omega$ , but also a state traversed during the execution of  $\omega$ , in which case  $\delta(s, \omega)$  is the remaining number of steps until termination. Let  $s_0 \in S$ ,  $\omega = \langle I, \pi, \beta \rangle$ , and  $d > 0$  be an integer. We start by considering the probability  $\mathbb{P}[\delta(s_0, \omega) = d]$ , which is given by:

$$\sum_{\bar{s} \in S^d} \sum_{\bar{a} \in A^d} \mathbb{P}[s_0, a_0, s_1, a_1, \dots, a_{d-1}, s_d, \perp] \text{ ,}$$

where  $\bar{s} = s_1 \dots s_d$  is the sequence of states traversed by  $\omega$ ,  $\bar{a} = a_0 \dots a_{d-1}$  is a sequence of actions chosen by  $\pi_{\omega}$ , and  $\perp$  denotes the option termination. Note the appearance of  $\perp$  at the end explicitly requires the option to last for exactly  $d$  steps; state trajectories which stop earlier are not

<sup>1</sup>We could be more precise and restrict ourselves to *discrete phase-type distributions* because these are enough to model the duration until absorption events in Markov chains; but this does not simplify our arguments at this point. The class of distributions we consider will become clear in Section 3.1

considered. By the Markov property,

$$\begin{aligned} & \mathbb{P}[s_0, a_0, \dots, s_{d-1}, a_{d-1}, s_d, \perp] \\ &= \left( \prod_{i=0}^{d-1} \mathbb{P}[a_i, s_{i+1} | s_i] \cdot \mathbb{P}[\# | s_{i+1}] \right) \cdot \frac{\mathbb{P}[\perp | s_d]}{\mathbb{P}[\# | s_d]} \\ &= \left( \prod_{i=0}^{d-1} \pi(s_i, a_i) \mathbb{P}(s_i, a_i, s_{i+1}) (1 - \beta(s_{i+1})) \right) \cdot \frac{\beta(s_d)}{1 - \beta(s_d)}. \end{aligned}$$

With some algebra, it can be shown that summing this expression over  $\bar{s}$  and  $\bar{a}$  yields:

$$\mathbb{P}[\delta(s_0, \omega) = d] = \mathbf{e}_{s_0}^\top \mathbf{A}_{\omega, \#}^{d-1} \mathbf{A}_{\omega, \perp} \mathbf{1}, \quad (1)$$

where  $\mathbf{e}_{s_0} \in \mathbb{R}^S$  is an indicator vector with  $\mathbf{e}_{s_0}(s) = \mathbb{I}[s = s_0]$ ,  $\mathbf{A}_{\omega, \#} \in \mathbb{R}^{S \times S}$  is a matrix with  $\mathbf{A}_{\omega, \#}(s, s') = \sum_{a \in A} \pi(s, a) P(s, a, s') (1 - \beta(s'))$ ,  $\mathbf{A}_{\omega, \perp} \in \mathbb{R}^{S \times S}$  is a matrix with  $\mathbf{A}_{\omega, \perp}(s, s') = \sum_{a \in A} \pi(s, a) P(s, a, s') \beta(s')$ , and  $\mathbf{1} \in \mathbb{R}^S$  is a vector of ones.

Using the ODM notation introduced in the previous section, Equation (1) can be written as

$$\mathbb{P}_{s_0}[d | \omega] = \mathbf{e}_{s_0}^\top \mathbf{A}_{\omega, \#}^{d-1} \mathbf{A}_{\omega, \perp} \mathbf{1}.$$

Now we would like to extend this expression to sequences containing more than one option. Using a similar derivation, for any  $t > 0$ ,  $\bar{d} \in \mathbb{N}^t$ , and  $\bar{\omega} \in \Omega^t$ , we can compute  $\mathbb{P}_{s_0}[\bar{d} | \bar{\omega}]$  as follows:

$$\mathbf{e}_{s_0}^\top \mathbf{A}_{\omega_1, \#}^{d_1-1} \mathbf{A}_{\omega_1, \perp} \mathbf{A}_{\omega_2, \#}^{d_2-1} \mathbf{A}_{\omega_2, \perp} \dots \mathbf{A}_{\omega_t, \#}^{d_t-1} \mathbf{A}_{\omega_t, \perp} \mathbf{1}.$$

Note that the same reasoning applies if we consider distributions over states. That is, instead of a fixed initial state  $s_0$ , we consider a state sampled according to some distribution  $\alpha_0$  over  $S$ . In that case, we define the random variable  $\delta(\alpha_0, \omega)$  representing the duration of option  $\omega$  when started from  $s_0 \sim \alpha_0$ . By the same argument as above,

$$\mathbb{P}_{\alpha_0}[d | \omega] = \mathbb{P}[\delta(\alpha_0, \omega) = d] = \boldsymbol{\alpha}_0^\top \mathbf{A}_{\omega, \#}^{d-1} \mathbf{A}_{\omega, \perp} \mathbf{1},$$

where  $\boldsymbol{\alpha}_0 \in \mathbb{R}^S$  is the vector representation of  $\alpha_0$ . Again, this can be extended in the same way to sequences with more than one option. Therefore, we have proved the following result.

**Theorem 1.** *Let  $M$  be an MDP with  $n$  states,  $\Omega$  a set of options, and  $\Sigma = \Omega \times \{\#, \perp\}$ . For every distribution  $\alpha_0$  over the states of  $M$ , there exists a PSR  $\mathcal{A} = \langle \boldsymbol{\alpha}_0, \mathbf{1}, \{\mathbf{A}_\sigma\} \rangle$  with at most  $n$  states that computes the distributions over durations of options executed from a state sampled according to  $\alpha_0$ .*

Note that the MDP transition kernel and the options' stochastic policies are coupled inside the transition matrices of PSR  $\mathcal{A}$  representing the ODM. This coupling is the reason why we can model the timing of options in an MDP via a process with observation on the set  $\Omega \times \{\#, \perp\}$  whose

size is *independent* of the set of primitive actions  $A$ , and whose transitions operators have size at most  $|S|$ . Note that this can be particularly interesting in settings where the number of possible actions is very large but a small number of options is enough to specify the ‘‘useful’’ modes of operation of an agent.

### 3.2 ODM for Explorable Trajectories

Note that the ODM representation  $\mathcal{A}$  given by Theorem 1 can be used to query the probability of observing any trajectory in  $(\Omega \times \{\#, \perp\})^*$  starting from a state sampled from  $\alpha$ . In principle, this includes trajectories which are not valid for an agent interacting with an MDP via options – e.g. we can query the probability of trajectories of the form  $(\omega_1, \#)^{d_1} (\omega_2, \#)^{d_2} \dots$ , where  $\omega_1$  was interrupted before termination. Note that this type of trajectories will never be observed by an agent that explores an environment by interacting with it only via options executed in call-and-return fashion. In particular, an agent does not need to learn about these trajectories if the goal is to plan via options only, and cannot hope to learn about these trajectories if it only explores the environment via options. We now show that a PSR representation for an ODM can be restricted to produce non-zero probabilities for legal trajectories only, without increasing the size of the model too much.

Recall that  $\Sigma = \Omega \times \{\#, \perp\}$ . When options are always executed to termination, valid trajectories belong to a subset of  $\Sigma^*$ :

$$V = \left( \bigcup_{\omega \in \Omega} \{(\omega, \#)^*(\omega, \perp)\} \right)^*.$$

That is, each sequence of option-continuation events must end with an option-termination event before a new option can be started.

Now we want to modify the model in Theorem 1 so that it only assigns non-zero probabilities to trajectories in  $V$ . Let  $f$  be the function computed by  $\mathcal{A}$ . Then we want another PSR computing the function given by  $\tilde{f}(x) = f(x)$  for  $x \in V$  and 0 otherwise. We now show that  $\tilde{f}$  can also be computed by a PSR with size close to that of  $\mathcal{A}$ .

**Theorem 2.** *If  $\mathcal{A}$  has  $n$  states, then there exists a PSR  $\tilde{\mathcal{A}}$  with at most  $(|\Omega| + 1)n$  states computing  $\tilde{f}$ .*

*Proof.* By the Carlyle–Paz–Fliess theorem, it suffices to show that  $\text{rank}(\tilde{\mathbf{H}}) \leq (|\Omega| + 1)n$ , where  $\tilde{\mathbf{H}} \in \mathbb{R}^{\Sigma^* \times \Sigma^*}$  is the bi-infinite Hankel matrix associated with  $\tilde{f}$ . Let  $\chi_V : \Sigma^* \rightarrow \mathbb{R}$  be the characteristic function of the regular language  $V$ , which is given by  $\chi_V(x) = 1$  if  $x \in V$  and  $\chi_V(x) = 0$  otherwise. Note that it is easy to construct a DFA with  $|\Omega| + 1$  states recognizing  $V$ . Such a DFA can also be written as a PSR with  $|\Omega| + 1$  states computing the function  $\chi_V$ . Thus, the rank of the Hankel matrix  $\mathbf{H}_V$  associated with  $\chi_V$  is at most  $|\Omega| + 1$ . Now note that we

can write  $\tilde{\mathbf{H}} = \mathbf{H} \circ \mathbf{H}_V$ , where  $\mathbf{H}$  is the Hankel matrix of  $f$  and  $\circ$  denotes entry-wise multiplication (also known as Hadamard matrix product). Finally, we use an elementary bound on the rank of Hadamard products to conclude that

$$\text{rank}(\tilde{\mathbf{H}}) \leq \text{rank}(\mathbf{H}) \cdot \text{rank}(\mathbf{H}_V) = (|\Omega| + 1)n . \quad \square$$

Note that the bound on the number of states of  $\tilde{\mathcal{A}}$  is in general not tight. For example, in our experiments with a grid-like environment, we observed that starting with an MDP with  $n$  states the true rank of  $\tilde{\mathbf{H}}$  is on the order of  $O(\sqrt{n})$ . So the PSR representation of ODM can lead to more compact models than the underlying MDP, and can potentially lead to faster learning and planning algorithms. This issue should be investigated further in future work.

## 4 SPECTRAL LEARNING OF ODM

The PSR representation of ODM given by the theorems in previous section can be computed explicitly if the underlying MDP is known exactly. In this section we explore the possibility of learning such a representation from data about option durations, without exploiting any knowledge about the underlying MDP. The rationale behind this approach is that it can lead to more compact representations that do not depend on the set of primitive actions, and learn only the parts of the MDP’s state space which are reachable via options.

Because an option duration model over valid trajectories can be represented with a PSR of moderate size, we can use the spectral learning algorithm in [Boots *et al.*, 2011] to estimate an ODM from a set of trajectories in  $\Sigma^*$  produced by the agent exploring the MDP using a fixed policy over options. For each trajectory, the initial state is sampled according to a fixed distribution  $\alpha$ . We assume that the options executed by the agent are selected according to some fixed open-loop policy. This is important if we want to use the sampled trajectories to learn a model of the environment which is independent of the exploration policy.

The algorithm takes as input  $\Sigma$  and a basis  $\mathcal{B}$  in  $\Sigma^*$ , uses them to estimate the corresponding Hankel matrices, and then recovers a PSR by performing singular value decomposition and linear algebra operations on these matrices. Although the method works almost out-of-the-box, in practice the results tend to be sensitive to the choice of basis. Thus, after briefly recapitulating how the spectral learning algorithm proceeds, we will devote the rest of the section to describe a procedure for building a basis which is tailored for the case of learning option duration models.

Suppose the basis  $\mathcal{B}$  is fixed and the desired number of states  $n$  is given. Suppose that a set of sampled trajectories was used to estimate the Hankel matrices  $\mathbf{H}, \mathbf{H}_\sigma \in \mathbb{R}^{\mathcal{P} \times \mathcal{S}}$  and vectors  $\mathbf{h}_\mathcal{P} \in \mathbb{R}^{\mathcal{P}}, \mathbf{h}_\mathcal{S} \in \mathbb{R}^{\mathcal{S}}$  defined in Sec. 2.3. The algorithm starts by taking the truncated SVD  $\mathbf{U}_n \mathbf{D}_n \mathbf{V}_n^\top$

of  $\mathbf{H}$ , where  $\mathbf{D}_n \in \mathbb{R}^{n \times n}$  contains the first  $n$  singular values of  $\mathbf{H}$ , and  $\mathbf{U}_n \in \mathbb{R}^{\mathcal{P} \times n}$  and  $\mathbf{V}_n \in \mathbb{R}^{\mathcal{S} \times n}$  contain the first left and right singular vectors respectively. Finally, we compute the transition operators of a PSR as  $\mathbf{A}_\sigma = \mathbf{D}_n^{-1} \mathbf{U}_n^\top \mathbf{H}_\sigma \mathbf{V}_n$ , and the initial and final weights as  $\alpha_\lambda^\top = \mathbf{h}_\mathcal{S}^\top \mathbf{V}_n$  and  $\alpha_\infty = \mathbf{D}_n^{-1} \mathbf{U}_n^\top \mathbf{h}_\mathcal{P}$ . This yields a PSR with  $n$  states. It was proved in [Boots *et al.*, 2011] this algorithm is statistically consistent: if the population Hankel matrices are known and the basis  $\mathcal{B}$  is complete, then the learned PSR is equivalent to the one that generated the data.

### 4.1 Basis Selection

Finding a complete basis for a given function  $f : \Sigma^* \rightarrow \mathbb{R}$  is in general a hard combinatorial problem: although if  $\mathbf{H}_f$  has rank  $n$ , there exists a complete basis with  $n$  prefixes and suffixes, the search space where these lie is doubly-exponential in  $n$ . To overcome this problem, randomized and greedy basis selection heuristics have been proposed in the past [Wiewiora, 2005; Balle *et al.*, 2012]. Greedy selection is in general computationally expensive because it requires re-learning the model every time a new element is added to the basis. Randomized search does not work in our case because it cannot take into account the fact that we have syntactic constraints on the valid trajectories. Therefore, we designed a procedure that takes the structure of our problem into account.

Our procedure is parametrized by various quantities depending on the environment and the observed trajectories: the maximum possible duration  $T_\omega$  of each option  $\omega \in \Omega$  in our training dataset, and  $T = \max_\omega T_\omega$ ; an upper bound  $K_r \geq 1$  on the number of option executions needed to reach every possible state in  $M$  when initial states are sampled from  $\alpha$ ; and, an upper bound  $K_d \geq 1$  on the number of option executions needed to distinguish/disambiguate every pair of states in  $M$  in terms of option duration information.

We assume these quantities are given, either because they can be derived from prior knowledge of the application domain, or because they are cross-validated. The definitions imply that  $T$  and  $K_r$  are a function of the geometry of the environment and how this relates to the available options. On the other hand,  $K_d$  measures the statistical similarity between states in  $M$  when seen through the lens of  $\Omega$ . The intuition is simple: we want to ensure that we have enough prefixes in the Hankel matrix to get to all reachable states in  $M$ , and enough suffixes to make sure each of these states can be distinguished from each other. The following construction formalizes this intuition.

We obtain the set  $\mathcal{P}$  of prefixes in the basis as the union of

two disjoint sets. The first set is:

$$\mathcal{P}_\perp = \{x_1 x_2 \cdots x_t \mid 1 \leq t \leq K_r, x_i = (\omega_i, \#)^{d_i}(\omega_i, \perp), \omega_i \in \Omega, 0 \leq d_i \leq T_{\omega_i}\} .$$

These are trajectories with at most  $K_r$  option executions, containing all possible sequences of options, and all possible option durations allowed by the model. A simple calculation shows that  $|\mathcal{P}_\perp| = O(K_r T^{K_r} |\Omega|^{K_r})$ . Note that each trajectory in  $\mathcal{P}_\perp$  terminates with a symbol  $(\omega, \perp)$ . If we remove this last symbol for each trajectory, we obtain a disjoint set of prefixes  $\mathcal{P}_\# = \{x \mid x(\omega, \perp) \in \mathcal{P}_\perp\}$ . Then we take  $\mathcal{P} = \mathcal{P}_\perp \cup \mathcal{P}_\#$ . Note that again we have  $|\mathcal{P}| = O(K_r T^{K_r} |\Omega|^{K_r})$ .

Similarly, the set of suffixes will be obtained as the union of two sets. These are defined as follows:

$$\begin{aligned} \mathcal{S}_\# &= \{x_1 x_2 \cdots x_t \mid 1 \leq t \leq K_s, x_i = (\omega_i, \#)^{d_i}(\omega_i, \perp), \\ &\quad \omega_i \in \Omega, 0 \leq d_i \leq T_{\omega_i}\} , \\ \mathcal{S}_\perp &= \{(\omega, \perp)x \mid x \in \mathcal{S}_\#, \omega \in \Omega\} . \end{aligned}$$

The suffixes in  $\mathcal{S}_\#$  are obtained like the prefixes in  $\mathcal{P}_\perp$ , with the only difference that the number of option executions is now upper bounded by  $K_s$  instead of  $K_r$ . Suffixes in  $\mathcal{S}_\perp$  are obtained by prefixing each string in  $\mathcal{S}_\#$  with each possible option termination symbol  $(\omega, \perp)$ . The whole set of suffixes is  $\mathcal{S} = \mathcal{S}_\perp \cup \mathcal{S}_\#$ , and we have  $|\mathcal{S}| = O(K_s T^{K_s} |\Omega|^{K_s+1})$ .

Note that not every string in  $\mathcal{PS}$  defines a valid trajectory. This is required for the Hankel matrices  $\mathbf{H}_\sigma$  to be different from zero; otherwise the operators  $\mathbf{A}_\sigma$  cannot be correctly recovered. To see why this is the case, consider the basis  $\mathcal{B}' = (\mathcal{P}_\perp, \mathcal{S}_\#)$ . By construction we have  $\mathcal{P}_\perp \mathcal{S}_\# \subset V$ . However, if we consider a system where some  $\omega$  never lasts for just one step, then every trajectory in  $\mathcal{P}_\perp \{(\omega, \perp)\} \mathcal{S}_\#$  has probability zero. In particular, in such a system the matrix  $\mathbf{H}_\sigma$  over the basis  $\mathcal{B}'$  is exactly zero. To work around this problem, it is necessary to introduce non-valid trajectories in the basis, to ensure that  $\mathbf{H}$  will contain some sub-blocks filled with zeros, but the  $\mathbf{H}_\sigma$  will contain some non-zero sub-blocks.

## 4.2 Scalability

Though at first glance the exponential dependence on  $K_r$  and  $K_s$  in the size of the basis may seem prohibitive, in practice the situation is much better. On one hand, these are worst-case bounds that assume all possible sequences of options and durations need to be considered. However, in practice some of those might never be observed — e.g. because the duration of certain options has little variance, or because some options are not available in many configurations. On the other hand, as already mentioned, we have prior knowledge indicating that a significant fraction of the entries appearing in the resulting Hankel matrices will

be zero because they correspond to non-valid trajectories. This will lead to highly sparse matrices which can benefit from sparse representations, sparse singular value decompositions, and sparse numerical linear algebra in general. In those cases, the complexity of the operations involved will not depend on the size of these matrices, but instead on the number of non-zero entries. This quantity will in practice be much smaller than the size of the Hankel matrices given by our choice of basis, and we have observed such benefits in our experiments.

## 5 PLANNING WITH ODM

In this section we want to briefly investigate the potential utility of ODM representations for planning with options. Specifically, we want to show how to use ODMs as a form of state representation for value function learning. In a nutshell, our main result states that nothing is lost by planning with options using a PSR representation of an ODM instead of the original MDP representation.

The first important observation is that the construction in Theorem 1 (and in consequence, the construction in Theorem 2) leads to a PSR with identical transition operators  $\{\mathbf{A}_\sigma\}$  regardless of which initial distribution over the states of  $M$  is chosen. That is, the only thing that needs to be modified when we change the initial distribution over the states is the initial state in the PSR. Thus, given an MDP  $M$  and a set of options  $\Omega$ , we can fix the transition structure of a PSR representation for the corresponding ODM — either over all trajectories or only valid trajectories (the distinction is not important for the purpose of this section). These will be given by  $\{\mathbf{A}_\sigma\}$  for  $\sigma \in \Sigma = \Omega \times \{\#, \perp\}$ , and the corresponding final weights  $\alpha_\infty$ . We denote by  $n'$  the dimension of this PSR. Then we can map each possible distribution  $\alpha$  over the states of  $M$  to a PSR state  $\theta_\alpha \in \mathbb{R}^{n'}$ . Note that although this mapping is straightforward for the construction in Theorem 1, this is not the case for the construction in Theorem 2, and even less for the PSR recovered by the learning algorithm in Section 4 because such a PSR is only equivalent to the original up to conjugation of the transition weights by an invertible matrix [Boots *et al.*, 2011]. In addition to distributions over initial states, we can also consider PSR states obtained by the state-update procedure. That is, if we are given a valid trajectory  $u \in V$  representing a history of options and associated continuation and termination events, then we can obtain the updated state

$$\theta_{\alpha, u}^\top = \frac{\theta_\alpha^\top \mathbf{A}_u}{\theta_\alpha^\top \mathbf{A}_u \alpha_\infty} .$$

This represents another probability distribution over the states of  $M$ . Note that by construction we have  $\theta_{\alpha, \lambda} = \theta_\alpha$ . With this notation we can show that the state-option value function of any policy over options is linear in the PSR state.

**Theorem 3.** Let  $\pi_\Omega : S \times \Omega \rightarrow [0, 1]$  be a stochastic stationary policy over options on the MDP  $M$ . For every  $\omega \in \Omega$  there exists a vector  $\rho_\omega \in \mathbb{R}^{n'}$  so that for every distribution  $\alpha$  over states in  $M$  and every history  $u \in V$ , we have  $\mathbb{E}_s[Q^{\pi_\Omega}(s, \omega)] = \theta_{\alpha, u}^\top \rho_\omega$ , where the expectation is over states  $s$  sampled from the distribution induced by observing  $u$  after starting in a state drawn from  $\alpha$ .

The above theorem includes as a special case the situation in which the agent is at a fixed state in  $M$ ; i.e. no uncertainty. The proof follows along the lines of a similar result known for PSRs with reward structure induced by a corresponding POMD [James *et al.*, 2004]. The key difference is that we need to take into account the semi-Markov nature of option termination events, which make the computations involved in the discount factors and extended transition kernels more involved. Though the details are not complicated, the algebra is too lengthy to be included in this version and will be presented in full in an extended version.

Although planning with PSRs has been studied using methods akin to POMDP planning [James *et al.*, 2004; Izadi and Precup, 2008; Boots *et al.*, 2011], we chose a more efficient alternative, the Fitted-Q Iteration (FQI) algorithm of Ernst *et al.* [2005]. Similar uses of FQI with PSR states have been proposed in Ong *et al.* [2012]; Hamilton *et al.* [2013]. FQI learns a policy through an iterative re-fitting process over batches of sampled transitions of the form:  $\langle s, a, r, s' \rangle$ . At every iteration, an ensemble of extremely randomized trees (ExtraTrees) [Geurts *et al.*, 2006] is fit to  $Q(s, a)_t = r + \gamma \max_{a'} Q_{t-1}(s', a')$  over all quadruples  $s, a, r, s'$ . In order to ensure convergence, the tree structure must be kept fixed across iterations. Hence, we fit the regressor in the first iteration and only refreshed the values and not the structure on subsequent iterations. We plan directly over the ODM state vector updated at each step with the corresponding operator (continuation or termination). So, in the first step, we compute  $\theta_0 = \alpha_\lambda^\top \mathbf{A}_{\sigma_0}$ , then update  $\theta_t = \theta_{t-1}^\top \mathbf{A}_{\sigma_t}$ . The ODM state vector is then normalized in order to prevent difficulties with the regressor. This approach is leveraging a well-known planning approach, so it is straightforward. It is possible that the structure of ODM can be exploited further to design more efficient planning algorithms. We leave this topic for future work.

## 6 EXPERIMENTS

We first assess the performance of the spectral learning algorithm with our data-driven basis construction method for  $K_r = 2$  and  $K_s = 1$ . We use a 4-connected grid with four actions representing the cardinal directions (NEWS). Unless the current state is a “wall” each action moves the agent one step in the specified direction with probability 0.9, and maintains the current state with probability 0.1. We also define one option for each cardinal direction. These options

take as many steps as possible in the specified direction until they hit a wall, at which point the option terminates. A uniformly random exploration policy is used for sampling 10000 episodes in which five options are executed to termination. We also collected a test set consisting of 10000 trajectories of 8 options sequences. We evaluate the prediction accuracy by computing the relative error over the estimated remaining number of steps in the currently executing option. For each test trajectory, we picked a time index uniformly at random and conditioned the learned ODM on the history up to this point. These random split points were then kept fixed throughout all evaluations. Figure 1b shows that the prediction accuracy increases with the dimension of the ODM. More samples also allow for better predictions. Since the prediction task is inherently stochastic, even the true ODM cannot achieve 0 relative error.

### 6.1 Planning in a grid world

We formulate the grid-world problem by giving a reward of +100 for entering the bottom-right state and -10 per collision, with discount factor 0.9. A dataset of 1000 trajectories of 8 options each was collected with a uniformly random policy over options. For each curve in figure 1, we use our dataset to simultaneously learn an ODM and plan over it. We evaluate the performance of the greedy policy by taking 100 Monte-Carlo estimates in the simulated environment. Given the true underlying MDP and a set of options, we compute the resulting Semi-Markov Decision Process (SMDP) (see p. 26 of Sutton *et al.* [1999]) and solve it using value iteration, to obtain the baseline (optimal achievable return) for our evaluation. We extended the ExtraTrees implementation of Pedregosa *et al.* [2011] with the freezing mechanism required for FQI. We used all available features for splitting internal nodes and fitted an ensemble of 10 trees with maximum depth of 5. Figure 1c shows that an optimal policy can be obtained using 1000 trajectories and one planning step.

### 6.2 Planning with a simulated robot

Using the same methodology, we experimented with the proposed ODM learning and planning approach in a simulated robot environment with continuous state space, non-linear dynamics, and actions that correspond to real actuators of the robot. We leverage the simulation capabilities of the 2D physics engine Box2D<sup>2</sup> to obtain realistic accelerations, collisions and friction effects.

We set the density of a circular differential wheeled robot to 3.0 and its radius to 0.17 cm. A primitive action in this domain consists in the application of force vector of (0.1, 0.1) on both wheels every 1/10 of a simulated second. At the beginning of every episode, the robot is initialized at position (0.5, 0.5) of a 2x2 meters environment.

<sup>2</sup><http://box2d.org>



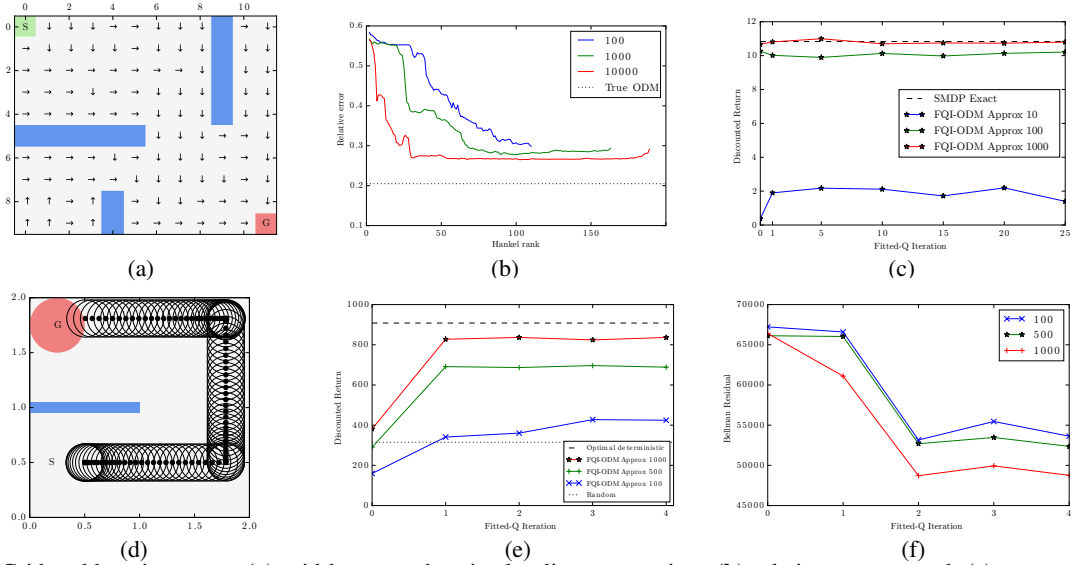


Figure 1: Gridworld environment: (a) grid layout and optimal policy over options (b) relative error vs rank (c) average discounted cumulative return. Simulated robot environment: (d) trajectory of an optimal policy (e) average discounted cumulative reward (f) mean square Bellman residual

A 10cm thick wall separates the robot from the goal location at  $(0.25, 0.75)$ . The episode finishes when the robot is within 25cm of the target.

As in the grid-world problem, we define a set of options over four radial directions. The termination condition triggers when the front-facing distance sensor detects an obstacle 6cm ahead. Because of the nonlinear dynamics, the hand-defined options controller does not always stop exactly within this distance. Given enough acceleration, the robot might sometimes touch the walls. This additional source of noise due to an imperfect controller is interesting from an experimental point of view. A stochastic component also makes the forward action ineffective 5% of the time. The rewards are 1000 for reaching the goal region and -10 by collision. Taking a primitive action incurs no cost but there is a discount factor  $\gamma = 0.999$ .

We collected 2000 trajectories of at most 10 options with a uniformly random policy over options. Due to the size of the environment, we found that trajectories of at most 5 options were insufficient to consistently learn good policies. We used  $K_r = 2, K_s = 1$  for the basis, considered all features for node splitting and used 10 trees of maximum depth 8. The results presented in Figure 1e were obtained by resampling 10 times the original dataset without replacement, to characterize the stability of the learning algorithm. The greedy policy derived from FQI was then evaluated with 10 Monte-Carlo rollouts. Since the underlying MDP is unknown, we could not compute the exact optimal policy, so we upper-bounded the achievable return through the policy in Figure 1d under 100% success rate for the forward command. We used the average return of a uniform random policy as baseline. As expected,

the mean square Bellman residual [Ernst *et al.*, 2005] decreases for larger sample sizes and over longer planning horizon (Fig. 1f). While 100 episodes yield ODM policies slightly better than random, 1000 episodes are sufficient to recover a close-to-optimal solution (Fig. 1e).

## 7 DISCUSSION

The approach we presented learns a predictive model for option durations, and we illustrates its use in robot navigation tasks. As discussed, timing models are simple, yet in many problems they are sufficient for good quality planning. To see why having simpler models might be useful, consider the prevalence of bandit problems in ad placement; the task could be done better with full MDPs, data efficiency is more important for the application. Similarly, we believe that being able to exploit simple yet efficient timing models is interesting and important. The use of this type of model in planning algorithms should be investigated further. Models of states in terms of what happens after executing certain actions are known to be useful, e.g. Dayan [1993]. But our models are simplified, hence different both from action-respecting embeddings Bowling *et al.* [2005, 2007] and predictive state representations with options Wolfe and Singh [2006], which aim to learn full observation models conditioned on extended actions, in order to characterize the current state. Timing models get around the problem of both large action spaces (by using a finite set of options) and the problem of large observation spaces (by focusing only on continuation and termination). A theoretical analysis of the error of planning with timing models instead of true transition models is left for future work. We also aim to study the sample efficiency of this approach.

## References

- B. Balle, A. Quattoni, and X. Carreras. Local loss optimization in operator models: A new insight into spectral learning. *International Conference on Machine Learning (ICML)*, 2012.
- B. Boots, S. Siddiqi, and G. Gordon. Closing the learning planning loop with predictive state representations. *International Journal of Robotic Research*, 2011.
- M. Bowling, A. Ghodsi, and D. Wilkinson. Action respecting embedding. *International Conference on Machine Learning (ICML)*, 2005.
- M. Bowling, D. Wilkinson, A. Ghodsi, and A. Milstein. Subjective localization with action respecting embedding. *Robotics Research*, 2007.
- J. W. Carlyle and A. Paz. Realizations by stochastic finite automata. *Journal of Computer Systems Science*, 1971.
- P. Dayan. Improving generalisation for temporal difference learning: The successor representation. *Neural Computation*, 1993.
- W. Droste, M. Kuich and H. Vogler. *Handbook of weighted automata*. Springer, 2009.
- Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *J. Mach. Learn. Res.*, 6:503–556, December 2005.
- M. Fliess. Matrices de Hankel. *Journal de Mathématiques Pures et Appliquées*, 1974.
- Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Mach. Learn.*, 63(1):3–42, April 2006.
- J. J. Gibson. The theory of affordances. In R. Shaw and J. Bransford, editors, *Perceiving, Acting, and Knowing*, 1977.
- William L. Hamilton, Mahdi M. Fard, and Joelle Pineau. Modelling sparse dynamical systems with compressed predictive state representations. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, volume 28, pages 178–186, 2013.
- Masoumeh T. Izadi and Doina Precup. Point-based planning for predictive state representations. In *Advances in Artificial Intelligence*, volume 5032 of *Lecture Notes in Computer Science*, pages 126–137. Springer Berlin Heidelberg, 2008.
- H. Jaeger. Observable operator models for discrete stochastic time series. *Neural Computation*, 2000.
- Michael R James, Satinder Singh, and Michael L Littman. Planning with predictive state representations. In *Machine Learning and Applications, 2004. Proceedings. 2004 International Conference on*, pages 304–311. IEEE, 2004.
- M.L. Littman, R.S. Sutton, and S. Singh. Predictive representations of state. *Neural Information Processing Systems (NIPS)*, 2002.
- A. Machado, M. T. Malheiro, and W. Erlhagen. Learning to time: A perspective. *Journal of the Experimental Analysis of Behavior*, 2009.
- Sylvie C.W. Ong, Yuri Grinberg, and Joelle Pineau. Goal-directed online learning of predictive models. In Scott Sanner and Marcus Hutter, editors, *Recent Advances in Reinforcement Learning*, volume 7188 of *Lecture Notes in Computer Science*, pages 18–29. Springer Berlin Heidelberg, 2012.
- F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- M. L. Puterman. *Markov Decision Processes - Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., 1994.
- M. Rosencrantz, G. Gordon, and S. Thrun. Learning low dimensional predictive representations. *International Conference on Machine Learning (ICML)*, 2004.
- S. Singh, M. R. James, and M. R. Rudary. Predictive state representations: A new theory for modeling dynamical systems. *Uncertainty in Artificial Intelligence (UAI)*, 2004.
- J. Stober, R. Miikkulainen, and B. Kuipers. Learning geometry from sensorimotor experience. *Joint Conference on Development and Learning and Epigenetic Robotics*, 2011.
- R. S Sutton, D. Precup, and S. Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 1999.
- E. Wiewiora. Learning predictive representations from a history. *International Conference on Machine Learning (ICML)*, 2005.
- B. Wolfe and S. Singh. Predictive state representations with options. *International Conference on Machine Learning (ICML)*, 2006.

---

# Parameterizing the Distance Distribution of Undirected Networks

---

Christian Bauckhage

Fraunhofer IAIS and University of Bonn, Germany  
{fn.in}@iais.fraunhofer.de

Kristian Kersting and Fabian Hadiji

TU Dortmund University, Germany  
{fn.in}@cs.tu-dortmund.de

## Abstract

Network statistics such as node degree distributions, average path lengths, diameters, or clustering coefficients are widely used to characterize networks. One statistic that received considerable attention is the distance distribution — the number of pairs of nodes for each shortest-path distance — in undirected networks. It captures important properties of the network, reflecting on the dynamics of network spreading processes, and incorporates parameters such as node centrality and (effective) diameter. So far, however, no parameterization of the distance distribution is known that applies to a large class of networks. Here we develop such a closed-form distribution by applying maximum entropy arguments to derive a general, physically plausible model of path length histograms. Based on the model, we then establish the generalized Gamma as a three-parameter distribution for shortest-path distance in strongly-connected, undirected networks. Extensive experiments corroborate our theoretical results, which thus provide new approaches to network analysis.

## 1 INTRODUCTION

As networks are combinatorial structures, network analysis typically relies on statistics such as node degree distributions, average path lengths, clustering coefficients, or measures of assortativity [11]. One statistic that received considerable attention is the distance distribution — the number of pairs of nodes for each shortest-path distance — in undirected networks. First of all, features such as average path lengths or network diameters can be determined therefrom. Second of all, path length statistics are closely related to velocities or durations of network spreading processes. Analytically tractable models of shortest path distributions would thus allow for inferring network properties as well

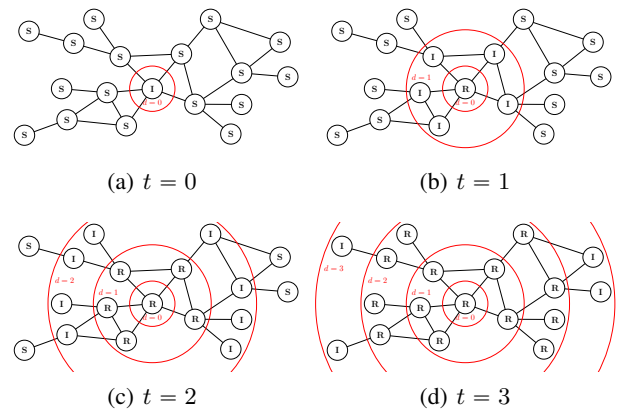


Figure 1: A highly infectious *SIR* cascade on a network realizes a node discovery process. (a) at onset time  $t = 0$ , most nodes are *susceptible* and only a single node is *infected*. (b)–(d) infected nodes immediately *recover* but always infect their susceptible neighbors. This way, the number of newly infected nodes  $n_t$  at time  $t$  corresponds to the number  $n_d$  of nodes that are at distance  $d = t$  from the source node of the epidemic.

as for reasoning about diffusion dynamics. Yet, analytic models and in particular parameterizing the distance distribution prove difficult to achieve and related reports are curiously scarce [7, 9, 14, 34, 35]. Here, we contribute to these efforts and derive a novel, principled general model of shortest path length distributions in strongly connected networks. Considering a duality between network spreading processes and shortest path lengths, we then show that maximum entropy arguments as to diffusion dynamics lead to the generalized Gamma distribution as a three-parameter distribution in closed form.

The work presented here was motivated by questions as to the dynamics of network spreading processes. Models of such processes play an important role in various disciplines. They model the dynamics of epidemic diseases [20, 27, 22], explain the diffusion of innovations or viral messages [1, 24], and, in the wake of social media, a

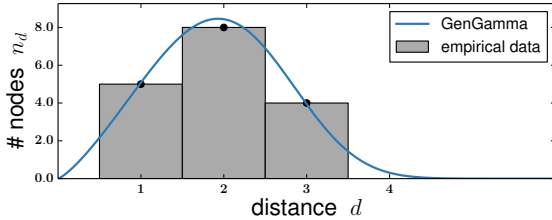


Figure 2: Shortest path histogram resulting from the network spreading process in Fig. 1 and a corresponding maximum likelihood fit of the generalized Gamma distribution.

quickly growing body of research develops graph diffusion models to study patterns of information dissemination in Web-based social networks [2, 10, 16, 25, 36]. Each of these examples concerns an instance of a rather general phenomenon: An agent (a virus, a rumor, an urge to buy a product, etc.) spreads in form of a contact process and thus cascades through a network of interlinked entities (people, computers, blogs, etc.). At the onset of the agent’s activity, many networked entities are *susceptible* to its effects but only few are actually *infected* (see Fig. 1(a)). As time progresses, susceptible entities related to infected ones may become infected whereas infected entities may remain infected, *recover*, become susceptible again, or even be removed from the population (see Fig. 1(b)–(d)). Crucial properties of such a process are its infection rate, its recovery rate, or the number of infected entities per unit of time. If an (information) epidemic is observed to rage through a community, these features help assessing its progression or final outbreak size and thus inform contagion or dissemination strategies. For instance, while public health authorities need to devise immunization protocols to curtail epidemic diseases, viral marketers typically aim at maximizing the effects of their campaigns. In any case, knowledge as to the structure of a community through which an epidemic spreads would be beneficial. Alas, in practice, community structures are hardly ever known but available information only consists of outbreak data as shown in Fig. 2.

Relating such outbreak data to network structures is the key idea underlying our contribution and is orthogonal to related approaches such as [2, 21, 26, 15, 36], which assume outbreak data *and* information about network members to be available and apply machine learning to identify hubs, link structures, or infection routes. Instead, we follow the paradigm in [4, 7, 16, 28, 29, 35] which asks for physical explanations of the noticeably skewed appearance of outbreak histograms and attempts to relate corresponding physical models to network properties.

More precisely, we make two technical contributions. First, we relate outbreak data to network structures via a novel maximum entropy model. Then, based on the model, we establish the generalized Gamma distribution as a physically-

plausible, continuous parameterization of the distance distribution of networks of arbitrary topology. Considering the fact that temporal distributions of infection counts in highly infectious spreading processes and distributions of shortest path lengths are dual phenomena allows us to invoke maximum entropy arguments from which we derive physical characterizations of path lengths- and outbreak statistics. In other words, our model results from first principles rather than from data mining. It also generalizes previous theoretical results [7, 35] and provides an explanation for a recent empirical observation as to path length distributions in social networks [8].

We proceed as follows. We start off by briefly reviewing existing analytical models of shortest path- and outbreak distributions in Section 2. Afterwards, we show that the generalized Gamma distribution naturally generalizes these, and discuss its properties and characteristics in Section 3. In particular, we demonstrate how the generalized Gamma emerges as a maximum entropy model of path length- and outbreak data. Before concluding, we support our theoretical results by exhaustive experiments on both synthetic and real-world graphs in Section 4.

## 2 KNOWN RESULTS

Analytically tractable models of shortest path length distributions and outbreak data are of considerable interest in the study of epidemic processes on networks. However, as networks are combinatorial structures, corresponding results are necessarily statistical [4, 7, 16, 28, 29, 35]. In a landmark paper [35], Vazquez studied the dynamics of epidemic processes in power law networks. Arguing based on branching process models, he showed that for networks whose node degree distribution has a power law exponent  $2 < \gamma < 3$ , the number of infected nodes at time  $t$  follows a *Gamma distribution*.

$$f_{GA}(t; \theta, \eta) = \frac{1}{\theta^\eta} \frac{1}{\Gamma(\eta)} t^{\eta-1} e^{-t/\theta} \quad (1)$$

where  $\Gamma(\cdot)$  is the gamma function and  $\theta > 0$  and  $\eta > 0$  are scale and shape parameters, respectively. Curiously, for power law exponents  $\gamma \geq 3$ , this result does not apply. Concerned with Erdős-Rényi graphs, Bauchhage *et al.* [7] derived a different result. Based on models of the expected number of paths of length  $t$  between arbitrary nodes [9, 14], they resorted to extreme value theory [13] to show that infection counts can be characterized by the *Weibull distribution*.

$$f_{WB}(t; \lambda, \kappa) = \frac{\kappa}{\lambda^\kappa} t^{\kappa-1} e^{-(t/\lambda)^\kappa} \quad (2)$$

where  $\lambda > 0$  and  $\kappa > 0$  are scale and shape parameters. Neither model was obtained from mere empirical observations. Both follow from basic principles, provide *physically and hence socially plausible* and comprehensible characterizations of shortest path length distributions and diffusion dynamics, and were verified empirically.

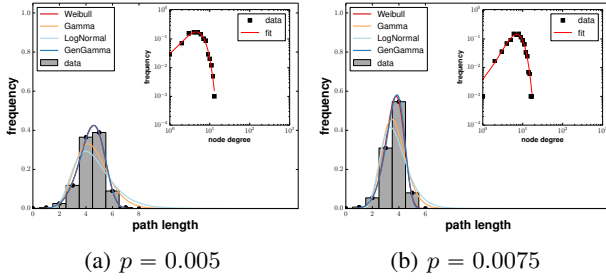


Figure 3: Qualitative examples of shortest path distribution in Erdős-Rényi graphs. Confirming [7], Weibull fits well. Yet, generalized Gamma provides better fits.

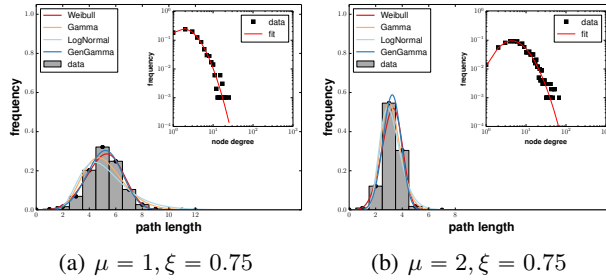


Figure 4: Qualitative examples of shortest path distribution in LogNormal graphs. For different node degree distributions, the models considered here provide more or less accurate fits. The generalized Gamma, however, always fits best (cf. Tab. 1).

Finally, Bild *et al.* [8] recently investigated information cascades and retweet networks on twitter and found that they could accurately fit their data using the *LogNormal distribution*

$$f_{LN}(t; \mu, \xi) = \frac{1}{\sqrt{2\pi} \xi t} e^{-\frac{(\log t - \mu)^2}{2\xi^2}} \quad (3)$$

where  $\mu$  and  $\xi$  are location and scale parameters. Regarding our approach in this paper, we emphasize that (3) was not found theoretically but emerged empirically. At the same time, we note that the Gamma, Weibull, and LogNormal distribution may easily confused for one another [31]. Our discussion so far therefore begs the question if the above results are contradictory or if they can be unified within a more general framework? In the following, we will show that the latter is indeed the case.

### 3 A THREE-PARAMETER DISTANCE DISTRIBUTION

We will now establish the *generalized Gamma distribution* as a comprehensive model of the shortest path distributions and outbreak data in connected networks of arbitrary topology. Different versions of the generalized Gamma dis-

tribution can be traced back to 1920s [12]. Here, we are concerned with the three-parameter version that was introduced by Stacy [33]. Its probability density function is defined for  $t \in [0, \infty)$  and given by

$$f_{GG}(t | \sigma, \alpha, \beta) = \frac{\beta}{\sigma^\alpha} \frac{1}{\Gamma(\alpha/\beta)} t^{\alpha-1} e^{-(t/\sigma)^\beta} \quad (4)$$

where  $\sigma > 0$  determines scale and  $\alpha > 0$  and  $\beta > 0$  are shape parameters. The probability density function in (4) is unimodal but may be skewed to the left or to the right. It also contains several other distributions as special cases [5, 12]. Most notably, we point out that setting  $\beta = 1$  yields the Gamma distribution in (1), equating  $\alpha = \beta$  yields the Weibull distribution in (2), letting  $\alpha/\beta \rightarrow \infty$ , the generalized Gamma distribution approaches the LogNormal distribution in (3).

These properties immediately suggest that the above results might be specific instances of a more general model. We will now prove that this is indeed the case:

**Theorem 1.** *The generalized Gamma distribution provides a physically plausible model of distance distributions and outbreak data in strongly-connected, undirected networks.*

Before proving this in the following subsections, we would like to stress that, in contrast to Vazquez [35] and Bauckhage *et al.* [7], our maximum entropy model does not make any assumption on the topology of the network — next to being strongly connected — through which a viral agent is spreading. Rather, it considers general properties of highly infectious processes as discussed in the introduction and resorts to the maximum entropy principle together with likelihood maximization techniques. Any aspects related to topological properties (e.g. degree distribution or clustering patterns) of individual networks are absorbed into the parameters of (4), which, as we shall see in the experimental section, is flexible enough to represent a wide range of different path length and spreading statistics.

#### 3.1 A MAXIMUM ENTROPY APPROACH

Let us now demonstrate that the generalized Gamma distribution provides a principled model of shortest path- and outbreak statistics in networks. We argue based on Jaynes’ *maximum entropy principle* [17]. It states that, subject to observations and contextual knowledge, the probability distribution that best represents the available information is the one of highest entropy. In particular, we resort to an approach by Wallis, cf. [18, chapter 11], which does not assume entropy as an a priori measure of uncertainty but uncovers it in the course of the argument.

To derive our analytical model of path length histograms of arbitrary networks, we consider the network spreading process in Fig. 1. Borrowing terminology from epidemiology, we note that, at onset time  $t = 0$ , most nodes in the network are *susceptible* and one node is *infected*. At time  $t + 1$ ,

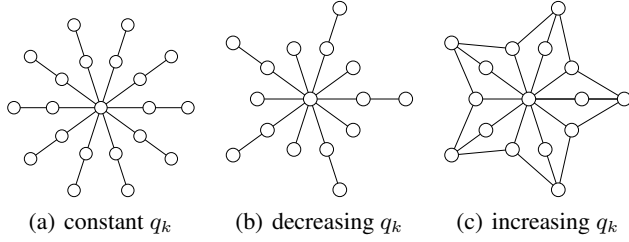


Figure 5: The probability of finding a path of length  $k$  may be constant or decrease or increase with  $k$ .

nodes that were infected at  $t$  have *recovered* yet did infect their susceptible neighbors. Highly infectious *SIR* cascades like this now realize a node discovery process: the number  $n_t$  of newly infected nodes at time  $t$  corresponds to the number of nodes that are at topological distance  $d = t$  from the source.

Given these considerations, we observe that the discrete shortest path distribution of the entire network is nothing but the sum of all node count histograms  $h_s[t]$  taken over all possible source nodes. Let  $K$  denote the length of the longest shortest path starting at a source  $v_s$  and use  $n_k$  to indicate the number of nodes that are  $k$  steps away from  $v_s$ . Then,  $n = \sum_{k=0}^K n_k$  where  $n$  denotes the total number of nodes. The probability of observing a node at distance  $k$  can thus be expressed as  $p_k = n_k/n$  and we have  $1 = \sum_{k=0}^K p_k$ . Moreover, we let  $q_k$  denote the probability that there exists a shortest path of length  $k$  so that  $1 = \sum_{k=0}^K q_k$  and we emphasize that  $p_k$  and  $q_k$  will generally differ (see the didactic examples in Fig. 5). With these definitions at hand, the joint probability of observing counts  $n_k$  of infected nodes corresponds to the multinomial

$$P(n_1, \dots, n_K) = n! \prod_{k=1}^K \frac{q_k^{n_k}}{n_k!}$$

whose log-likelihood is given by

$$L = \log n! + \sum_{k=0}^K n_k \log q_k - \log n_k! \quad (5)$$

Assuming that  $n_k \gg 1$ , we may apply Stirling's formula  $\log n_k! \approx n_k \log n_k - n_k$  to simplify (5)

$$\begin{aligned} L &\approx n \log n - n + \sum_{k=0}^K n_k (\log q_k - \log n_k + 1) \\ &= n \log n + n \sum_{k=0}^K p_k (\log q_k - (\log p_k + \log n)) \\ &= -n \sum_{k=0}^K p_k \log \frac{p_k}{q_k}. \end{aligned} \quad (6)$$

As the expression in (6) is a Kullback-Leibler divergence, our considerations so far led indeed to an entropy that needs to be maximized in order to determine the most likely values  $n_k^*$  of the  $n_k$ .

However, up until now, the quantities  $q_k$  are not defined precisely enough to allow for a solution. Also, (6) accounts

only indirectly for temporal aspects of network spreading processes as any dependency on time is hidden in the index of summation  $k$ . We address both these issues by choosing the ansatz

$$q_k = A t_k^{\alpha-1} \quad (7)$$

where  $A$  is a normalization constant and  $\alpha > 0$ . This choice and its significance will be justified in detail below. For now, we continue with our main argument.

Another underspecified quantity so far is the length  $K$  of the supposed longest shortest path. However, dealing with networks of finite size, we can bypass the need of having to specify  $K$  by means of introducing the following constraint

$$\sum_{k=0}^{\infty} n_k = n \quad (8)$$

into the problem of maximizing (6). Finally, to prevent degenerate solutions (e.g. instantaneous or infinite spread), we impose a constraint on the times  $t_k$  and require their moments

$$\sum_{k=0}^{\infty} \frac{n_k}{n} t_k^\beta = c \quad (9)$$

to be finite for some  $\beta > 0$ . Using differential forms, we now express the log-likelihood in (6) and the two constraints in (8) and (9) as

$$dL = \sum_{k=0}^{\infty} \frac{\partial L}{\partial n_k} dn_k = \sum_{k=0}^{\infty} (\log A t_k^{\alpha-1} - \log n_k) dn_k,$$

$dn = \sum_{k=0}^{\infty} dn_k$ , and  $dc = \sum_{k=0}^{\infty} t_k^\beta dn_k$ , and consider the Lagrangian with multipliers  $\rho$  and  $\gamma$

$$\mathcal{L} = \sum_{k=0}^{\infty} \left[ \log \frac{A t_k^{\alpha-1}}{n_k} - \rho - \gamma t_k^\beta \right] dn_k = 0$$

in order to determine most likely infection counts  $n_k^*$  for the spreading process described above. Since at the solution the bracketed terms  $[\cdot]$  must vanish identically for every  $dn_k$ , we immediately obtain

$$n_k^* = A e^{-\rho} t_k^{\alpha-1} e^{-\gamma t_k^\beta}.$$

Plugging this result back into  $p_k = n_k/n$  yields a discrete probability mass function in the now exp-transformed space

$$\frac{n_k^*}{n} = \frac{t_k^{\alpha-1} e^{-\gamma t_k^\beta}}{\sum_{j=0}^{\infty} t_j^{\alpha-1} e^{-\gamma t_j^\beta}} \quad (10)$$

which explicitly relates infection counts  $n_k$  to time steps  $t_k$ . However, (10) still depends on the Lagrangian multiplier  $\gamma$  (now in exp-space), which is not immediately related to any available data. In order to determine  $\gamma$ , we assume the duration  $\Delta t = t_{k+1} - t_k$  of time steps to be small. This permits the following approximation

$$\sum_{k=0}^{\infty} t_k^{\alpha-1} e^{-\gamma t_k^\beta} \Delta t \approx \int_0^{\infty} t^{\alpha-1} e^{-\gamma t^\beta} dt = \gamma^{-\frac{\alpha}{\beta}} \frac{\Gamma(\alpha/\beta)}{\beta}$$

so that

$$\frac{n_k^*}{n} = \Delta t \frac{\gamma^{\frac{\alpha}{\beta}} \beta}{\Gamma(\alpha/\beta)} t_k^{\alpha-1} e^{-\gamma t_k^\beta}.$$

Plugging this into (9), tedious but straightforward algebra yields  $\gamma = \frac{\alpha}{\beta c}$  which is to say that

$$\frac{n_k^*}{n} = \Delta t \left[ \frac{\beta}{\Gamma(\alpha/\beta)} \left( \frac{\alpha}{\beta c} \right)^{\frac{\alpha}{\beta}} \right] t_k^{\alpha-1} e^{-\frac{\alpha}{\beta c} t_k^\beta}. \quad (11)$$

In order to establish the final result, we now let  $\Delta t \rightarrow 0$  which leads to

$$\frac{n_k^*}{n} \approx \int_{\Delta t} f(\tau) d\tau \approx \Delta t f(t) \quad (12)$$

where  $t_k - \frac{\Delta t}{2} \leq t \leq t_k + \frac{\Delta t}{2}$ . Direct comparison of (11) and (12) together with the substitution  $\sigma = \left(\frac{\beta c}{\alpha}\right)^{1/\beta}$  finally establishes that

$$f(t) = \frac{\beta}{\sigma^\alpha} \frac{1}{\Gamma(\alpha/\beta)} t^{\alpha-1} e^{-(t/\sigma)^\beta}$$

which is indeed the generalized Gamma distribution that was introduced in (4).

### 3.2 EXTENSIONS TO DIFFERENT TIME SCALES

So far, our derivation above was based on properties of networked *SIR* cascades for which the infection rate  $i$  as well as the recovery rate  $r$  were both assumed to be 100%. Yet, the empirical result in Section 4 indicate that the generalized Gamma distribution also accounts for the dynamics of less infectious spreading processes where  $i$  and  $r$  are smaller. Such epidemics usually last longer as it takes more time to reach all susceptible nodes and infected nodes may remain so over extended periods. In other words, such processes can be thought of as taking place on a different time scale. Here, we briefly show that the generalized Gamma distribution can also explain spreading processes on linearly or polynomially transformed time scales.

Recall that if a random variable  $X$  is distributed according to  $f(x)$ , the monotonously transformed random variable  $Y = h(X)$  has a probability function that is given by

$$g(y) = f(h^{-1}(y)) \cdot \left| \frac{d}{dy} h^{-1}(y) \right|.$$

Now, if  $t$  is generalized Gamma distributed and  $\tau = ct$  is a linearly transformed version of  $t$ , then

$$t = \frac{\tau}{c} \quad \text{and} \quad \frac{dt}{d\tau} = \frac{1}{c}$$

so that  $\tau$  is distributed according to

$$\begin{aligned} g(\tau) &= \frac{\beta}{\sigma^\alpha} \frac{1}{\Gamma(\alpha/\beta)} \left( \frac{\tau}{c} \right)^{\alpha-1} e^{-(\tau/c\sigma)^\beta} \cdot \frac{1}{c} \\ &= \frac{\beta}{\sigma'^\alpha} \frac{1}{\Gamma(\alpha'/\beta')} \tau^{\alpha'-1} e^{-(\tau/\sigma')^\beta} \end{aligned}$$

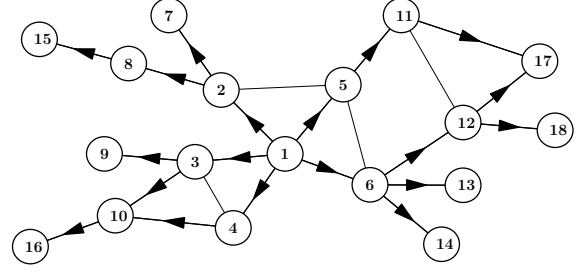


Figure 6: Causal graph of the process in Fig. 1. The adjacency matrix of a directed acyclic graph with a single source node can be brought into strictly upper triangular form and is thus nilpotent.

which is a generalized Gamma distribution with scale parameter  $\sigma' = c\sigma$ .

By the same token, if  $t$  is generalized Gamma distributed and  $\tau = t^c$  is a polynomially transformed version of  $t$ , then

$$t = \tau^{\frac{1}{c}} \quad \text{and} \quad \frac{dt}{d\tau} = \frac{1}{c} \tau^{\frac{1}{c}-1}$$

so that  $\tau$  is distributed according to

$$\begin{aligned} g(\tau) &= \frac{\beta}{\sigma^\alpha} \frac{1}{\Gamma(\alpha/\beta)} \left( \tau^{\frac{1}{c}} \right)^{\alpha-1} e^{-(\tau^{1/c}/\sigma)^\beta} \cdot \frac{1}{c} \tau^{\frac{1}{c}-1} \\ &= \frac{\beta' c}{\sigma'^{\alpha'}} \frac{1}{\Gamma(\alpha'/\beta')} \tau^{\alpha'-1} e^{-(\tau^{\beta'}/\sigma'^{\beta'})} \end{aligned}$$

which is a generalized Gamma distribution with parameters  $\sigma' = \sigma^c$ ,  $\alpha' = \frac{\alpha}{c}$ , and  $\beta' = \frac{\beta}{c}$ .

### 3.3 JUSTIFYING THE POLYNOMIAL ANSATZ (7)

While the constraints in (8) and (9) are arguably intuitive, the ansatz in (7) merits further elaboration. Note that *causal graphs* as in Fig. 6 indicate possible transmission pathways of an epidemic. In general they may be arbitrarily complex but, for the type of spreading process studied here, they are necessarily directed and acyclic. Now, in a directed, acyclic graph, the number  $N_k$  of shortest paths of lengths  $k$  starting from a source node  $v_s$  may indeed grow exponentially for a while (for instance in a tree) but the finite size of the graph causes  $N_k$  to drop to zero once  $k$  exceeds the length  $K$  of the longest shortest path; there are no shortest paths longer than  $K$ . Because of this finite size effect, a polynomial upper bound always exists for  $N_k$ ,  $k \leq K$ .

More formally, consider a network of  $n$  nodes. If  $v_s$  is the source node of an epidemic on this network and  $\mathbf{A}$  is the adjacency matrix<sup>1</sup> of the corresponding causal graph then  $(\mathbf{A}^k)_{s_j}$  denotes number of ways the epidemic agent can reach  $v_j$  from  $v_s$  in  $k$  steps and  $q_k \propto \sum_j (\mathbf{A}^k)_{s_j}$ . For causal

<sup>1</sup> $A_{ij} = 1$  if nodes  $v_i$  infects  $v_j$ , and  $A_{ij} = 0$  otherwise.

graphs,  $\mathbf{A}$  is strictly upper triangular and therefore *nilpotent*. That is,  $\exists K \leq n : \mathbf{A}^k = \mathbf{0} \ \forall k > K$ . Hence,  $\rho(\mathbf{A}) < 1$  which is to say  $\exists u > 0 : \|\mathbf{A}^k\|_F < u \ \forall k$  and therefore  $0 \leq (\mathbf{A}^k)_{sj} \leq \|\mathbf{A}^k\|_F < u$ . Accordingly,  $\sum_j (\mathbf{A}^k)_{sj} \in O(k^{\alpha-1})$  for some  $\alpha \geq 1$ . Setting  $t_k = \epsilon k$  where  $\epsilon \leq 1$  yields  $q_k \in O(t_k^{\alpha-1})$  which justifies (7).

Put in simple terms, it is sufficient to model the probability  $q_k$  of observing an epidemic path of length  $k$  as a polynomial in  $t$  rather than, say, an exponential function.

The fact that the parameter  $\alpha$  reflects aspects of network topology is easily seen from the examples in Fig. 5. Assuming the central node to be the source node of an epidemic the figure shows that  $q_k$  may be constant ( $\alpha = 1$ ), decreasing ( $\alpha < 1$ ), or increasing ( $\alpha > 1$ ) with  $k$ .

## 4 EMPIRICAL SUPPORT

We now support our theoretical results with empirical evidence. First, we present results on distance distributions of synthesized networks. Then, we report on experiments with a large number of spreading processes on synthetic graphs and, finally, we discuss results obtained for large, real-world networks.

Throughout, we fit continuous distributions to discrete histograms. That is, we apply functions  $f(t; \theta)$  to represent counts  $n_0, \dots, n_K$  which are grouped into  $K$  distinct intervals  $(t_0, t_1], (t_1, t_2], \dots, (t_{K-1}, t_\infty)$ . For this setting, it is advantageous to use multinomial likelihood estimation based on reweighted least squares in order to determine optimal model parameters  $\theta^*$  [19]. For a recent detailed exposition of this robust technique, we refer to [6]. For the fitted models, we report quantitative goodness-of-fit results in terms of the Hellinger distance

$H(h[t], f[t]) = \frac{1}{2} \sqrt{\sum_t \left( \sqrt{h[t]} - \sqrt{f[t]} \right)^2}$  between discrete empirical data  $h[t]$  and a discretized model  $f[t]$  where

$$f[t] = \begin{cases} F(t + \frac{1}{2}) & \text{if } t = 0 \\ F(t + \frac{1}{2}) - F(t - \frac{1}{2}) & \text{if } 0 < t < K \\ 1 - F(t + \frac{1}{2}) & \text{if } t = K \end{cases}$$

and  $F(\cdot)$  is the corresponding cumulative density function. We note that the Hellinger distance is bound as  $0 \leq H \leq 1$ .

**Synthetic Networks.** We created different Erdős-Rényi (ER), Barabási-Albert (BA), power law (PL), and Log-Normal (LN) graphs of  $n \in \{5,000, 10,000\}$  nodes. ER graphs are a staple of graph theory and merit investigation. To create ER graphs, we used edge probability parameters  $\pi \in \{0.005, 0.0075, 0.01\}$ . BA and PL graphs represent networks that result from preferential attachment processes and are frequently observed in biological, social, and technical contexts. To create BA graphs, we considered attachment parameters  $m \in \{1, 2, 3\}$  and exponents

Table 1: Goodness of Fit (avg. Hellinger distances) for shortest path histograms of synthetic networks

| network | parameters            | $f_{WB}$ | $f_{GA}$ | $f_{LN}$ | $f_{GG}$ |
|---------|-----------------------|----------|----------|----------|----------|
| ER      | $\pi = 0.0050$        | 0.058    | 0.170    | 0.227    | 0.051    |
|         | $\pi = 0.0075$        | 0.046    | 0.164    | 0.205    | 0.041    |
| BA      | $m = 1$               | 0.039    | 0.066    | 0.135    | 0.011    |
|         | $m = 2$               | 0.017    | 0.125    | 0.170    | 0.015    |
|         | $m = 3$               | 0.015    | 0.128    | 0.171    | 0.015    |
| PL      | $\gamma = 2.1$        | 0.154    | 0.042    | 0.040    | 0.030    |
|         | $\gamma = 2.3$        | 0.132    | 0.032    | 0.046    | 0.024    |
|         | $\gamma = 2.5$        | 0.123    | 0.037    | 0.064    | 0.028    |
|         | $\gamma = 2.7$        | 0.101    | 0.044    | 0.089    | 0.028    |
|         | $\gamma = 2.9$        | 0.081    | 0.050    | 0.108    | 0.026    |
|         | $\gamma = 3.1$        | 0.070    | 0.093    | 0.137    | 0.051    |
| LN      | $\mu = 1, \xi = 0.75$ | 0.075    | 0.093    | 0.149    | 0.037    |
|         | $\mu = 1, \xi = 1.25$ | 0.082    | 0.067    | 0.113    | 0.029    |
|         | $\mu = 2, \xi = 0.75$ | 0.073    | 0.092    | 0.145    | 0.036    |
|         | $\mu = 2, \xi = 1.25$ | 0.079    | 0.062    | 0.102    | 0.026    |

of the vertex degree distributions of the PL graphs were drawn from  $\gamma \in \{2.1, 2.2, \dots, 3.2\}$ . LN graphs show log-normally distributed vertex degrees and reportedly characterize link structures within sub-communities on the web [30]. To synthesize LN graphs, parameters were chosen from  $\mu \in \{1, 1.5, \dots, 3\}$  and  $\xi \in \{0.25, 0.5, 0.75, 1\}$ . For each parametrization, we created 100 instances.

Table 1 summarizes average goodness of fit results for graphs of  $n = 10,000$  nodes and different topologies (for lack of space, we omit results for some of the parametrizations considered in our experiments). We observe that (i) in agreement with Bauchhage *et al.* [7], the Weibull distribution provides a well fitting model for the distance distribution in ER graphs; it outperforms the Gamma and the Log-Normal distribution; (ii) in agreement with Vazquez [35], the Gamma distribution provides a well fitting model for PL graphs where  $2 < \gamma < 3$ ; (iii) for PL graphs where  $\gamma \lesssim 2.2$ , the LogNormal fits well, too; (iv) for PL graphs where  $\gamma \geq 3$ , the Weibull fits better than the Gamma or the LogNormal; in this context, we note that BA graphs are power law graphs for which  $\gamma = 3$  [3]; (v) in any case, the generalized Gamma distribution provides the best fits in all cases while still being physically plausible.

Indeed, the latter is not surprising as the densities in (1)-(3) are functions of two parameters whereas the generalized gamma in (4) depends on three parameters and therefore offers greater flexibility in statistical model fitting. However, as proven above, the generalized gamma also follows from first principle and provides accurate fits across a wide variety of underlying network topologies, while the Gamma and the Weibull distribution apply to particular types of networks only. In turn and probably most important, the traditional reliance of investigating several distributions can be eliminated. Moreover, as we will demonstrate below, even without regularization, which is an interesting avenue for



future work, the parameters of the generalized Gamma indeed allow one to distinguish different graph classes.

**Spreading Processes.** Given the synthetic networks created above, we simulated *SIR* spreading processes where the infection rate varied in  $i \in \{0.5, 0.6, \dots, 0.9\}$  and the recovery rate was chosen from  $r \in \{0.5, 0.6, \dots, 0.9\}$ . For each network and each choice of  $i$  and  $r$ , we created 10 epidemics starting at randomly selected source nodes  $v_s$  and fitted the generalized Gamma to the resulting outbreak data.

Table 2 shows exemplary results obtained for PL graphs of 10,000 nodes; rows correspond to different power law exponents  $\gamma$  and columns indicate different choices of pairs  $(i, r)$ . Each panel plots the outbreak data of all the corresponding epidemics (grey dots), the corresponding empirical average taken over the individual outbreak distributions (black dots), as well as a generalized Gamma fit to these averages (blue curves). Visual inspection of these results suggests that the generalized Gamma distribution accounts very well for average outbreak dynamics in power law graphs. Though not shown in the table, this behavior was also observed for individual epidemics as well as for epidemics on other networks, hence, supporting our theoretical justification provided in Sec. 3.2.

**Real-World Networks.** The KONECT network collection [23] provides a comprehensive set of large scale, real-world network data freely available for research. Networks contained in this collection comprise (online) social networks where edges indicate social contacts or friendship relations, natural networks such as power grids or connections between airports, and bipartite networks such as typically found in the context of recommender systems. The sizes of these networks vary between  $O(10,000)$  to  $O(1,000,000)$  nodes and they show different node degree distributions and clustering coefficients. For further details, we refer to <http://konect.uni-koblenz.de/>.

Tables 3 through 5 summarize goodness-of-fit results obtained from fitting the models in (1), (2), (3), and (4) to hop count distributions of social, natural, and bipartite networks respectively. Again in agreement with the theoretical prediction in [35], we observe that the Gamma distribution provides accurate fits to path length distributions in social networks which are often reported to be power law networks with power law exponents  $2 < \gamma < 3$ . For the case of natural and bipartite networks, we find the LogNormal distribution to provide better fits than the Gamma or the Weibull distribution. We emphasize that this is an empirical finding which, to our knowledge, has not yet been justified theoretically. In this sense, the work presented in this paper can be seen as the first such justification because the LogNormal is obtained a limiting case of the generalized Gamma distribution which, as shown in Sec. 3.1, provides a physically plausible model of distance distributions in networks. In fact, just as in the previous subsections, the

Table 3: Goodness of Fit (Hellinger distances) for shortest path histograms of social networks

| network                     | $f_{WB}$ | $f_{GA}$ | $f_{LN}$ | $f_{GG}$ |
|-----------------------------|----------|----------|----------|----------|
| advogato                    | 0.110    | 0.014    | 0.056    | 0.012    |
| arenas email                | 0.044    | 0.036    | 0.074    | 0.008    |
| arenas pgp                  | 0.100    | 0.022    | 0.057    | 0.013    |
| ca AstroPh                  | 0.157    | 0.021    | 0.047    | 0.021    |
| ca cit HepPh                | 0.133    | 0.023    | 0.065    | 0.015    |
| ca cit HepTh                | 0.089    | 0.016    | 0.026    | 0.010    |
| catster                     | 0.101    | 0.023    | 0.032    | 0.016    |
| cfinder google              | 0.076    | 0.021    | 0.023    | 0.032    |
| cit HepPh                   | 0.178    | 0.037    | 0.065    | 0.032    |
| cit HepTh                   | 0.135    | 0.018    | 0.051    | 0.016    |
| dblp cite                   | 0.078    | 0.042    | 0.077    | 0.012    |
| dogster                     | 0.215    | 0.036    | 0.048    | 0.026    |
| elec                        | 0.046    | 0.057    | 0.096    | 0.020    |
| email EuAll                 | 0.363    | 0.331    | 0.116    | 0.232    |
| enron                       | 0.122    | 0.046    | 0.075    | 0.023    |
| facebook wosn links         | 0.186    | 0.022    | 0.037    | 0.018    |
| facebook wosn wall          | 0.172    | 0.025    | 0.048    | 0.021    |
| filmtipset friend           | 0.140    | 0.024    | 0.052    | 0.017    |
| gottron net all             | 0.075    | 0.038    | 0.070    | 0.029    |
| gottron net core            | 0.049    | 0.024    | 0.060    | 0.006    |
| hep th citations            | 0.122    | 0.011    | 0.036    | 0.009    |
| loc brightkite edges        | 0.193    | 0.027    | 0.035    | 0.059    |
| munmun digg reply           | 0.170    | 0.028    | 0.054    | 0.023    |
| munmun twitter social       | 0.132    | 0.089    | 0.079    | 0.071    |
| collaboration               | 0.137    | 0.097    | 0.074    | 0.045    |
| ucsocial                    | 0.084    | 0.012    | 0.056    | 0.008    |
| petster carnivore           | 0.120    | 0.116    | 0.119    | 0.117    |
| petster friendships cat     | 0.109    | 0.023    | 0.032    | 0.027    |
| petster friendships dog     | 0.206    | 0.036    | 0.048    | 0.026    |
| petster friendships hamster | 0.167    | 0.083    | 0.049    | 0.060    |
| petster hamster             | 0.081    | 0.011    | 0.033    | 0.007    |
| sap                         | 0.142    | 0.063    | 0.090    | 0.052    |
| slashdot threads            | 0.267    | 0.082    | 0.055    | 0.102    |
| slashdot zoo                | 0.192    | 0.026    | 0.052    | 0.032    |
| wikiconflict                | 0.134    | 0.014    | 0.055    | 0.014    |
| wikisigned k2               | 0.268    | 0.166    | 0.070    | 0.136    |
| wikisigned nontext          | 0.291    | 0.086    | 0.058    | 0.066    |
| avg.                        | 0.146    | 0.050    | 0.059    | 0.039    |

Table 4: Goodness of Fit (Hellinger distances) for shortest path histograms of natural networks

| network                  | $f_{WB}$ | $f_{GA}$ | $f_{LN}$ | $f_{GG}$ |
|--------------------------|----------|----------|----------|----------|
| arenas meta              | 0.090    | 0.036    | 0.025    | 0.029    |
| as caida20071105         | 0.277    | 0.160    | 0.050    | 0.113    |
| as20000102               | 0.065    | 0.016    | 0.054    | 0.005    |
| dbpedia similar          | 0.685    | 0.686    | 0.082    | 0.018    |
| eat                      | 0.012    | 0.075    | 0.115    | 0.003    |
| lasagne frenchbook       | 0.017    | 0.001    | 0.058    | 0.003    |
| openflights              | 0.139    | 0.025    | 0.040    | 0.022    |
| powergrid                | 0.745    | 0.745    | 0.150    | 0.016    |
| usairport                | 0.053    | 0.010    | 0.040    | 0.007    |
| sociopatterns infectious | 0.030    | 0.026    | 0.053    | 0.012    |
| topology                 | 0.130    | 0.021    | 0.055    | 0.019    |
| wordnet words            | 0.192    | 0.031    | 0.054    | 0.022    |
| wordnet                  | 0.133    | 0.188    | 0.214    | 0.131    |
| avg.                     | 0.198    | 0.155    | 0.076    | 0.031    |

Table 2: Outbreak data obtained from spreading processes in power law networks of 10,000 nodes

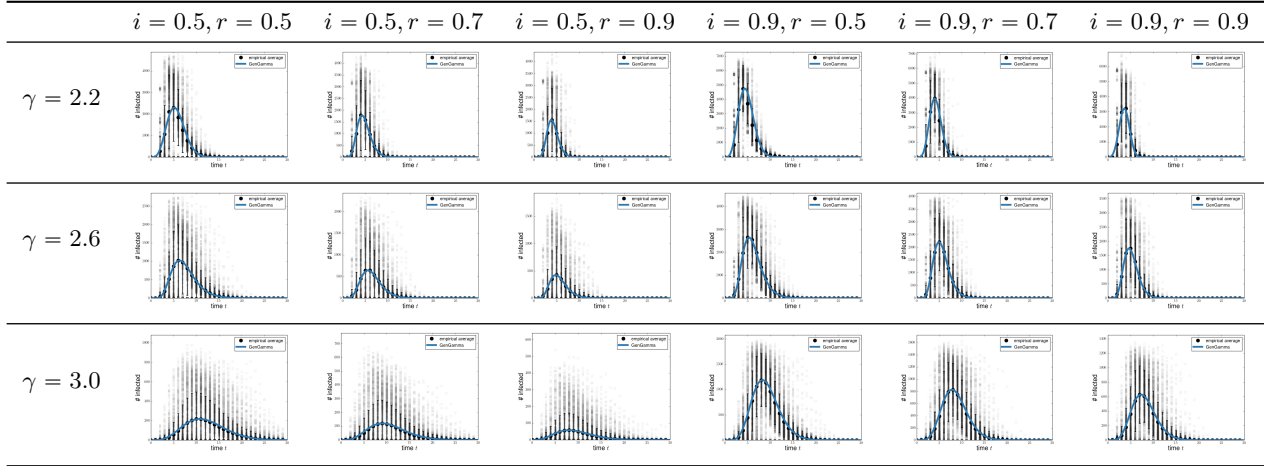


Table 5: Goodness of Fit (Hellinger distances) for shortest path histograms of bipartite networks

| network                  | $f_{WB}$ | $f_{GA}$ | $f_{LN}$ | $f_{GG}$ |
|--------------------------|----------|----------|----------|----------|
| adjnoun                  | 0.028    | 0.037    | 0.065    | 0.012    |
| bx                       | 0.264    | 0.096    | 0.090    | 0.130    |
| dbpedia occupation       | 0.165    | 0.103    | 0.113    | 0.101    |
| dbpedia producer         | 0.734    | 0.734    | 0.152    | 0.156    |
| dbpedia starring         | 0.721    | 0.721    | 0.034    | 0.032    |
| dbpedia writer           | 0.754    | 0.754    | 0.096    | 0.072    |
| epinions                 | 0.209    | 0.036    | 0.041    | 0.026    |
| escorts                  | 0.097    | 0.048    | 0.076    | 0.031    |
| filmtipset comment       | 0.060    | 0.049    | 0.089    | 0.007    |
| github                   | 0.186    | 0.078    | 0.082    | 0.078    |
| gottron reuters          | 0.128    | 0.112    | 0.137    | 0.106    |
| movielens 10m ti         | 0.187    | 0.082    | 0.101    | 0.074    |
| movielens 10m ui         | 0.036    | 0.083    | 0.116    | 0.031    |
| movielens 10m ut         | 0.199    | 0.146    | 0.159    | 0.144    |
| movielens 1m             | 0.043    | 0.005    | 0.040    | 0.007    |
| ucforum                  | 0.047    | 0.050    | 0.080    | 0.030    |
| pics ut                  | 0.268    | 0.164    | 0.166    | 0.183    |
| prosper support          | 0.230    | 0.259    | 0.232    | 0.208    |
| youtube groupmemberships | 0.192    | 0.115    | 0.119    | 0.115    |
| avg.                     | 0.239    | 0.193    | 0.105    | 0.081    |

generalized Gamma distribution is again found to provide the best overall fits to the data considered here. Qualitative examples of this behavior are shown in Fig. 7.

We also evaluated how the different distributions perform in predicting the average shortest path length and compared empirical means to the means of fitted models. For the Weibull, the mean is given by  $\lambda\Gamma(1 + 1/\kappa)$ , the mean of the Gamma is  $\eta\theta$ , that of the LogNormal is  $\exp(\mu - \xi^2/2)$ , and for the generalized Gamma we have  $\mathbb{E}\{t\} = \sigma \frac{\Gamma((\alpha+1)/\beta)}{\Gamma(\alpha/\beta)}$ . Using these formulas, we investigated predicted average shortest path lengths versus empirically determined ones for the networks in the KONECT collection. The results are summarized in Tab. 6, which lists mean squared square errors for the data in the figure. They suggests that w.r.t. predicting average path lengths, the Weibull performs worse

Table 6: Mean squared errors for predicted average shortest path lengths versus empirically determined ones for the networks in the KONECT collection.

|           | $f_{WB}$ | $f_{GA}$ | $f_{LN}$ | $f_{GG}$ |
|-----------|----------|----------|----------|----------|
| bipartite | 1.665    | 1.382    | 0.731    | 0.414    |
| natural   | 0.216    | 0.479    | 0.862    | 0.093    |
| social    | 0.474    | 0.219    | 0.648    | 0.220    |
| overall   | 1.745    | 1.479    | 1.303    | 0.478    |

than the Gamma which performs worse than the LogNormal which is outperformed by the generalized Gamma.

**Distinguishing between Different Graph Classes.** An interesting consequence of fitting the distance distribution using the three-parameter generalized Gamma is that it provides a non-linear mapping of path length data into three dimensions. This allows for visual analytics of the behavior of different graph topologies w.r.t. distance distributions. Fig. 8 shows exemplary distance distributions in terms 3D coordinates  $(\sigma, \alpha, \beta)$  that result from fitting generalized Gamma distributions. Looking at the Figure, it appears that distance distributions obtained from different network topologies cluster together or are confined to certain regions in this parameter space. These preliminary observations are arguably the most interesting finding in this paper as they suggest that the idea of characterizing networks in terms of continuous models of shortest path distributions can inform approaches to the problem of network inference from outbreak data. Investigating these results more deeply provides an interesting avenue for future work.

## 5 CONCLUSIONS

We considered the problem of parameterizing the distance distribution and epidemic outbreak data of strongly-connected, undirected networks. Invoking the maxi-

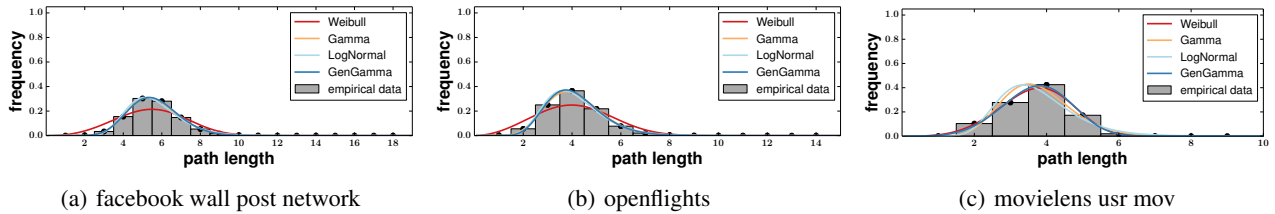


Figure 7: Qualitative examples of shortest path distributions on real-world networks. Path length distributions of social and natural networks are well accounted for by the Gamma distribution (a+b) whereas the Weibull provides better fits for bipartite networks that occur in recommender settings (c). In each case, however, the generalized Gamma fits best.

mum entropy principle, we showed that the generalized Gamma distribution provides a physically plausible, three-parameter distribution for these data, independent of the topology of the underlying networks. This result generalizes earlier models [7, 35] and explains recent empirical observations made w.r.t. twitter retweet networks [8]. Empirical tests confirmed our theoretical prediction and revealed that the generalized Gamma distribution accounts well for distance distributions of synthesized Erdős-Rényi, Barabási-Albert, power law, and LogNormal graphs as well as for real-world network in the KONECT collection [23]. Finally, we illustrated that the parameters of the generalized Gamma provide striking structural regularities in the resulting low-dimensional network representations.

Our results suggest several attractive avenues for future research. First of all, one should relate the shape and scale parameters of the generalized Gamma distribution to physical properties or well established features of networks. Second, one should use our theoretical results to devised informed sampling schemes for the problem of computing shortest paths (and their histograms) for large real-world networks. Finally, the results in Fig. 8 suggest a nearest-neighbour approach to network inference from outbreak data. Newly observed epidemic processes are matched to a large data base of 3D representations of outbreak data, for which the network is known; at least this provides helpful prior information for network inference approaches such as [32, 22].

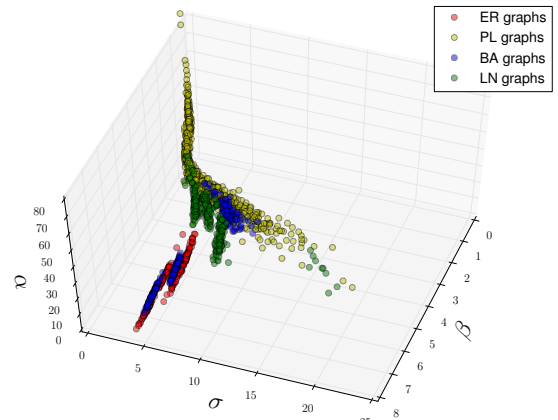
**Acknowledgments.** The authors would like to thank the anonymous reviewers for their feedback. The work was partly supported by the DFG Collaborative Research Center SFB 876 project A6.

## References

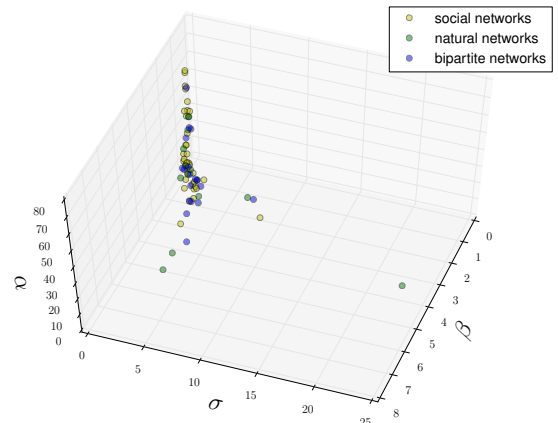
[1] D. Acemoglu, A. Ozdaglar, and M.E. Yildiz. Diffusion of Innovations in Social Networks. In *Proc. Int. Conf. on Decision and Control*. IEEE, 2011.

[2] E. Adar and A. Adamic. Tracking Information Epidemics in Blogspace. In *Proc. Int. Conf. on Web Intelligence*. IEEE/WIC/ACM, 2005.

[3] A.L. Barabasi and R. Albert. Emergence of Scaling



(a) synthetic networks



(b) real-world networks

Figure 8: 3D embedding of shortest path histograms from different networks. Each point  $(\sigma, \alpha, \beta)$  represents a path length distribution in terms of the parameters of the best fitting generalized Gamma model. Different classes of graphs appear to be confined to specific regions.

in Random Networks. *Science*, 286(5439):509–512, 1999.

[4] M. Barthelemy, A. Barrat, R. Pastor-Satorras, and A. Vespignani. Velocity and Hierarchical Spread of

- Epidemic Outbreaks in Scale-free Networks. *Physical Review Letters*, 92(17):178701, 2004.
- [5] C. Bauckhage. Computing the Kullback-Leibler Divergence between two Generalized Gamma Distributions. *arXiv:1401.6853 [cs.IT]*, 2014.
- [6] C. Bauckhage and K. Kersting. Strong Regularities in Growth and Decline of Popularity of Social Media Services. *arXiv:1406.6529 [cs.SI]*, 2014.
- [7] C. Bauckhage, K. Kersting, and B. Rastegarpanah. The Weibull as a Model of Shortest Path Distributions in Random Networks. In *Proc. Int. Workshop on Mining and Learning with Graphs*, 2013.
- [8] D.R. Bild, Y. Liu, R.P. Dick, Z. Morley Mao, and D.S. Wallach. Aggregate Characterization of User Behavior in Twitter and Analysis of the Retweet Graph. *ACM Trans. on Internet Technology*, 15(1):1–24, 2015.
- [9] V.D. Blondel, J.L. Guillaume, J.M. Hendrickx, and R.M. Jungers. Distance Distribution in Random Graphs and Application to Network Exploration. *Physical Review E*, 76(6):066101, 2007.
- [10] C. Budak, D. Agrawal, and A. El Abbadi. Limiting the Spread of Misinformation in Social Networks. In *Proc. WWW*. ACM, 2010.
- [11] R. Cohen and S. Havlin. *Complex Networks*. Cambridge University Press, 2010.
- [12] G.E. Crooks. The Amoroso Distribution. *arXiv:1005.3274 [math.ST]*, 2010.
- [13] L. de Haan and A. Ferreira. *Extreme Value Theory*. Springer, 2006.
- [14] A. Fronczak, P. Fronczak, and J.A. Holyst. Average Path Length in Random Networks. *Physical Review E*, 70(5):056110, 2004.
- [15] M. Gomez-Rodriguez, J. Leskovec, and B. Schölkopf. Structure and Dynamics of Information Pathways in Online Media. In *Proc. WSDM*. ACM, 2013.
- [16] J.L. Iribarren and E. Moro. Impact of Human Activity Patterns on the Dynamics of Information Diffusion. *Physical Review Letters*, 103(3):038702, 2009.
- [17] E.T. Jaynes. Information Theory and Statistical Mechanics. *Physical Review*, 106:620–630, 1957.
- [18] E.T. Jaynes. *Probability Theory: The Logic of Science*. Cambridge University Press, 2003.
- [19] R.I. Jennrich and R.H. Moore. Maximum Likelihood Estimation by Means of Nonlinear Least Squares. In *Proc. of the Statistical Computing Section*. American Statistical Association, 1975.
- [20] M.J. Keeling and K.T.D. Eames. Networks and Epidemic Models. *J. Royal Society Interface*, 2(4):295–307, 2005.
- [21] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the Spread of Influence through A Social Network. In *Proc. KDD*. ACM, 2003.
- [22] A. Kumar, D. Sheldon, and B. Srivastava. Collective diffusion over networks: Models and inference. In *Proc. UAI*, 2013.
- [23] J. Kunegis. KONECT: the Koblenz Network Collection. In *Proc. WWW*. ACM, 2013.
- [24] J. Leskovec, L.A. Adamic, and B.A. Huberman. The Dynamics of Viral Marketing. *ACM Trans. Web*, 1(1):5, 2007.
- [25] J. Leskovec, L. Backstrom, and J. Kleinberg. Meme-tracking and the Dynamics of the News Cycle. In *Proc. KDD*. ACM, 2009.
- [26] J. Leskovec, M. McGlohon, C. Faloutsos, N. Glance, and M. Hurst. Patterns of Cascading Behavior in Large Blog Graphs. In *Proc. Int. Conf on Data Mining*. SIAM, 2007.
- [27] A.L. Lloyd and R.M. May. How Viruses Spread Among Computers and People. *Science*, 292(5520):1316–1317, 2001.
- [28] M.E.J. Newman. The Spread of Epidemic Diseases on Networks. *Physical Review E*, 66(1):016128, 2002.
- [29] R. Pastor-Satorras and A. Vespignani. Epidemic Spreading in Scale-Free Networks. *Physical Review Letters*, 86(14):3200–3203, 2001.
- [30] D.M. Pennock, G.W. Flake, S. Lawrence, E.J. Glover, and C.L. Gilles. Winners Don’t Take All: Characterizing the Competition for Links on the Web. *PNAS*, 99(8):5207–5211, 2002.
- [31] H. Rinne. *The Weibull Distribution*. Chapman & Hall / CRC, 2008.
- [32] D. Sheldon, B.N. Dilkina, A.N. Elmachtoub, R. Finseth, A. Sabharwal, J. Conrad, C.P. Gomes, D.B. Shmoys, W. Allen, O. Amundsen, and W. Vaughan. Maximizing the spread of cascades using network design. In *In Proc. UAI*, pages 517–526, 2010.
- [33] E.W. Stacy. A Generalization of the Gamma Distribution. *The Annals of Mathematical Statistics*, 33(3):1187–1192, 1962.
- [34] A. Ukkonen. Indirect Estimation of Shortest Path Distributions with Small-World Experiments. In *Proc. Advances in Intelligent Data Analysis*, 2014.
- [35] A. Vazquez. Polynomial Growth in Branching Processes with Diverging Reproduction Number. *Physical Review Letters*, 96(3):038702, 2006.
- [36] J. Yang and J. Leskovec. Patterns of Temporal Variation in Online Media. In *Proc. WSDM*. ACM, 2011.

---

# New Limits for Knowledge Compilation and Applications to Exact Model Counting

---

**Paul Beame\***

Computer Science and Engineering  
University of Washington  
Seattle, WA 98195  
beame@cs.washington.edu

**Vincent Liew\***

Computer Science and Engineering  
University of Washington  
Seattle, WA 98195  
vliew@cs.washington.edu

## Abstract

We show new limits on the efficiency of using current techniques to make exact probabilistic inference for large classes of natural problems. In particular we show new lower bounds on knowledge compilation to SDD and DNNF forms. We give strong lower bounds on the complexity of SDD representations by relating SDD size to best-partition communication complexity. We use this relationship to prove exponential lower bounds on the SDD size for representing a large class of problems that occur naturally as queries over probabilistic databases. A consequence is that for representing unions of conjunctive queries, SDDs are not qualitatively more concise than OBDDs. We also derive simple examples for which SDDs must be exponentially less concise than FBDDs. Finally, we derive exponential lower bounds on the sizes of DNNF representations using a new quasipolynomial simulation of DNNFs by nondeterministic FBDDs.

## 1 Introduction

Weighted model counting is a fundamental problem in probabilistic inference that captures the computation of probabilities of complex predicates over independent random events (Boolean variables). Although the problem is  $\#P$ -hard in general, there are a number of practical algorithms for model counting based on DPLL algorithms and on knowledge compilation techniques. The knowledge compilation approach, though more space intensive, can be much more convenient since it builds a representation for an input predicate independent of its weights that allows the count to be evaluated easily given a particular choice of weights; that representation also can be reused to analyze more complicated predicates. Moreover,

with only a constant-factor increase in time, the methods using DPLL algorithms can be easily extended to be knowledge compilation algorithms [Huang and Darwiche, 2007]. (See [Gomes et al., 2009] for a survey.)

The representation to be used for knowledge compilation is an important key to the utility of these methods in practice; the best methods are based on restricted classes of circuits and on decision diagrams. All of the ones considered to date can be seen as natural sub-classes of the class of *Decomposable Negation Normal Form (DNNF)* formulas/circuits introduced in [Darwiche, 2001], though it is not known how to do model counting efficiently for the full class of DNNF formulas/circuits. One sub-class for which model counting is efficient given the representation is that of *d-DNNF* formulas, though there is no efficient algorithm known to recognize whether a DNNF formula is d-DNNF.

A special case of d-DNNF formulas (with a minor change of syntax) that is easy to recognize is that of *decision-DNNF* formulas. This class of representations captures all of the practical model counting algorithms discussed in [Gomes et al., 2009] including those based on DPLL algorithms. Decision-DNNFs include *Ordered Binary Decision Diagrams (OBDDs)*, which are canonical and have been highly effective representations for verification, and also *Free BDDs (FBDDs)*, which are also known as read-once branching programs. Using a quasipolynomial simulation of decision-DNNFs by FBDDs, [Beame et al., 2013, Beame et al., 2014] showed that the best decision-DNNF representations must be exponential even for many very simple 2-DNF predicates that arise in probabilistic databases.

Recently, [Darwiche, 2011] introduced another subclass of d-DNNF formulas called *Sentential Decision Diagrams (SDDs)*. This class is strictly more general than OBDDs and (in its basic form) is similarly canonical. (OBDDs use a fixed ordering of variables, while SDDs use a fixed binary tree of variables, known as a *vtree*.) There has been substantial development and growing application of SDDs to knowledge representation problems, including a recently released SDD software package [SDD, 2014]. In-

---

\*Research supported by NSF grant CCF-1217099.

deed, SDDs hold potential to be more concise than OBDDs. [Van den Broeck and Darwiche, 2015] showed that *compressing* an SDD with a fixed vtree so that it is canonical can lead to an exponential blow-up in size, but much regarding the complexity of SDD representations has remained open.

In this paper we show the limitations both of general DNNFs and especially of SDDs. We show that the simulation of decision-DNNFs by FBDDs from [Beame et al., 2013] can be extended to yield a simulation of general DNNFs by OR-FBDDs, the nondeterministic extension of FBDDs, from which we can derive exponential lower bounds for DNNF representations of some simple functions.

For SDDs we obtain much stronger results. In particular, we relate the SDD size required to represent predicate  $f$  to the “best-case partition” communication complexity [Kushilevitz and Nisan, 1997] of  $f$ . Using this, together with reductions to the communication complexity of disjointness (set intersection), we derive the following results:

- (1) There are simple predicates given by 2-DNF formulas for which FBDD size is polynomial but for which SDD size must be exponential.
- (2) For a natural, widely-studied class of database queries known as *Unions of Conjunctive Queries (UCQ)*, the SDD size is linear iff the OBDD size is linear and is exponential otherwise (which corresponds to a query that contains an *inversion* [Jha and Suciu, 2013]).
- (3) Similar lower bounds apply to the dual of UCQ, which consists of universal, positive queries.

To prove our SDD results, we show that for any predicate  $f$  given by an SDD of size  $S$ , using its associated vtree we can partition the variables of  $f$  between two players, Alice and Bob, in a nearly balanced way so that they only need to send  $\log^2 S$  bits of communication to compute  $f$ . The characterization goes through an intermediate step involving *unambiguous* communication protocols and a clever deterministic simulation of such protocols from [Yannakakis, 1991].

*Related work:* Beyond the lower bounds for decision-DNNFs in [Beame et al., 2013, Beame et al., 2014] which give related analyses for decision-DNNFs, the work of [Pipatsrisawat and Darwiche, 2010] on structured DNNFs is particularly relevant to this paper<sup>1</sup>. [Pipatsrisawat and Darwiche, 2010] show how sizes of what they term (*deterministic*)  $\mathbf{X}$ -*decompositions* can yield lower bounds on the sizes of *structured (deterministic)* DNNFs, which include SDDs as a special case. [Pipatsrisawat, 2010] contains the full details of how this can be applied to prove lower bounds for specific predicates. These bounds are actually equivalent to lower

<sup>1</sup>We thank the conference reviewers for bringing this work to our attention.

bounds exponential in the best-partition nondeterministic (respectively, unambiguous) communication complexity of the given predicates. Our paper derives this lower bound for SDDs directly but, more importantly, provides the connection to best-partition *deterministic* communication complexity, which allows us to have a much wider range of application; this strengthening is necessary for our applications.

**Roadmap:** We give the background and some formal definitions including some generalization required for this work in Section 2. We prove our characterization of SDDs in terms of best-partition communication complexity in Section 3 and derive the resulting bounds for SDDs for natural predicates in Section 4. We describe the simulation of DNNFs by OR-FBDDs, and its consequences, in Section 5.

## 2 Background and Definitions

We first give some basic definitions of DNNFs and decision diagrams.

**Definition 2.1.** A Negation Normal Form (NNF) *circuit* is a Boolean circuit with  $\neg$  gates, which may only be applied to inputs, and  $\vee$  and  $\wedge$  gates. Further, it is Decomposable (DNNF) iff the children of each  $\wedge$  gate are reachable from disjoint sets of input variables. (Following convention, we call this circuit a “DNNF formula”, though it is not a Boolean formula in the usual sense of circuit complexity.) A DNNF formula is deterministic (d-DNNF) iff the functions computed at the children of each  $\vee$  gate are not simultaneously satisfiable.

**Definition 2.2.** A Free Binary Decision Diagram (FBDD) is a directed acyclic graph with a single source (the root) and two specified sink nodes, one labeled 0 and the other 1. Every non-sink node is labeled by a Boolean variable and has two out-edges, one labeled 0 and the other 1. No path from the root to either sink is labeled by the same variable more than once. It is an OBDD if the order of variable labels is the same on every path. The Boolean function computed by an FBDD is 1 on input  $\mathbf{a}$  iff there is a path from the root to the sink labeled 1 so that for every node label  $X_i$  on the path,  $\mathbf{a}_i$  is the label of the out-edge taken by the path. An OR-FBDD is an FBDD augmented with additional nodes of arbitrary fan-out labeled  $\vee$ . The function value for the OR-FBDD follows the same definition as for FBDDs; the  $\vee$ -nodes simply make more than one path possible for a given input. (See [Wegener, 2000].)

We now define sentential decision diagrams as well as a small generalization that we will find useful.

**Definition 2.3.** For a set  $\mathbf{X}$ , let  $\top : \{0, 1\}^{\mathbf{X}} \rightarrow \{0, 1\}$  and  $\perp : \{0, 1\}^{\mathbf{X}} \rightarrow \{0, 1\}$  denote the constant 1 function and constant 0 function, respectively.

**Definition 2.4.** We say that a set of Boolean functions  $\{p_1, p_2, \dots, p_\ell\}$ , where each  $p_i$  has domain  $\{0, 1\}^{\mathbf{X}}$ , is dis-

joint if for each  $i \neq j$ ,  $p_i \wedge p_j = \perp$ . We call  $\{p_1, p_2, \dots, p_\ell\}$  a partition if it is disjoint and  $\bigvee_{i=1}^{\ell} p_i = \top$ .

**Definition 2.5.** A vtree for variables  $\mathbf{X}$  is a full binary tree whose leaves are in one-to-one correspondence with the variables in  $\mathbf{X}$ .

We define *Sentential Decision Diagrams* (SDDs) together with the Boolean functions they represent and use  $\langle \cdot \rangle$  to denote the mapping from SDDs to Boolean functions. (This notation is extended to sets of SDDs yielding sets of Boolean functions.) At the same time, we also define a directed acyclic graph (DAG) representation of the SDD.

**Definition 2.6.**  $\alpha$  is an SDD that respects vtree  $\mathbf{v}$  rooted at  $v$  iff:

- $\alpha = \top$  or  $\alpha = \perp$ .  
Semantics:  $\langle \top \rangle = \top$  and  $\langle \perp \rangle = \perp$ .  
 $G(\alpha)$  consists of a single leaf node labeled with  $\langle \alpha \rangle$ .
- $\alpha = X$  or  $\alpha = \neg X$  and  $v$  is a leaf with variable  $X$ .  
Semantics:  $\langle X \rangle = X$  and  $\langle \neg X \rangle = \neg X$   
 $G(\alpha)$  consists of a single leaf node labeled with  $\langle \alpha \rangle$ .
- $\alpha = \{(p_1, s_1), \dots, (p_\ell, s_\ell)\}$ ,  $v$  is an internal vertex with children  $v_L$  and  $v_R$ ,  $p_1, \dots, p_\ell$  are SDDs that respect the subtree rooted at  $v_L$ ,  $s_1, \dots, s_\ell$  are SDDs that respect the subtree rooted at  $v_R$ , and  $\langle p_1 \rangle, \dots, \langle p_\ell \rangle$  is a partition.  
Semantics:  $\langle \alpha \rangle = \bigvee_{i=1}^{\ell} (\langle p_i \rangle \wedge \langle s_i \rangle)$   
 $G(\alpha)$  has a circle node for  $\alpha$  labeled  $v$  with  $\ell$  child box nodes labeled by the pairs  $(p_i, s_i)$ . A box node labeled  $(p_i, s_i)$  has a left child that is the root of  $G(p_i)$  and a right child that is the root of  $G(s_i)$ . The rest of  $G(\alpha)$  is the (non-disjoint) union of graphs  $G(p_1), \dots, G(p_\ell)$  and  $G(s_1), \dots, G(s_\ell)$  with common sub-DAGs merged. (See Figure 1.)

Each circle node  $\alpha'$  in  $G(\alpha)$  itself represents an SDD that respects a subtree of  $\mathbf{v}$  rooted at some vertex  $v'$  of  $\mathbf{v}$ ; We say that  $\alpha'$  is in  $\alpha$  and use  $\text{Sdds}(v', \alpha)$  to denote the collection of  $\alpha'$  in  $\alpha$  that respect the subtree rooted at  $v'$ . The size of an SDD  $\alpha$  is the number of nodes in  $G(\alpha)$ .

Circle nodes in  $G(\alpha)$  may be interpreted as OR gates and paired box nodes may be interpreted as AND gates. In the rest of this paper, we will view SDDs as a class of Boolean circuit. The vtree property and partition property of SDDs together ensure that this resulting circuit is a d-DNNF.

We define a small generalization of vtrees which will be useful for describing SDDs with respect to a partial assignment of variables.

**Definition 2.7.** A pruned vtree for variables  $\mathbf{X}$  is a full binary tree whose leaves are either marked stub or by a variable in  $\mathbf{X}$ , and whose leaves marked by variables are in one-to-one correspondence with the variables in  $\mathbf{X}$ .

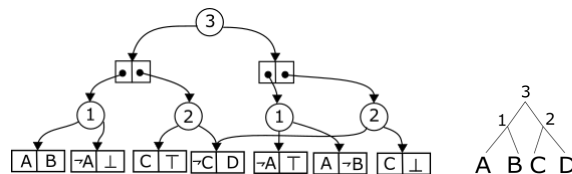


Figure 1: An SDD with its associated vtree that computes the formula  $(A \wedge B \wedge C) \vee (\neg C \wedge D)$

We generalize SDDs so that they can respect pruned vtrees. The definition is almost identical to that for regular SDDs so we only point out the differences.

**Definition 2.8.** The definition of a pruned SDD  $\alpha$  respecting a pruned vtree  $\mathbf{v}$ , its semantics, and its graph  $G(\alpha)$ , are identical to those of an SDD except that

- if the root vertex  $v$  of  $\mathbf{v}$  is a stub then  $\langle \alpha \rangle$  must be  $\perp$  or  $\top$ , and
- if the root vertex  $v$  of  $\mathbf{v}$  is internal then we only require that  $\langle p_1 \rangle, \dots, \langle p_\ell \rangle$  are disjoint but not necessarily that they form a partition.

We now sketch a very brief overview of the communication complexity we will need. Many more details may be found in [Kushilevitz and Nisan, 1997]. Given a Boolean function  $f$  on  $\{0, 1\}^{\mathbf{X}} \times \{0, 1\}^{\mathbf{Y}}$ , one can define two-party protocols in which two players, Alice, who receives  $x \in \{0, 1\}^{\mathbf{X}}$  and Bob, who receives  $y \in \{0, 1\}^{\mathbf{Y}}$  exchange a sequence of messages  $m_1, \dots, m_C = f(x, y) \in \{0, 1\}$  to compute  $f$ . (After each bit, the player to send the next bit must be determined from previous messages.) The (deterministic) communication complexity of  $f$ ,  $CC(f(\mathbf{X}, \mathbf{Y}))$ , is the minimum value  $C$  over all protocols computing  $f$  such that all message sequences are of length at most  $C$ . The one-way deterministic communication complexity of  $f$ ,  $CC_{\mathbf{X} \rightarrow \mathbf{Y}}(f(\mathbf{X}, \mathbf{Y}))$  is the minimum value of  $C$  over all protocols where Alice may send messages to Bob, but Bob cannot send messages to Alice.

For nondeterministic protocols, Alice simply guesses a string based on her input  $x$  and sends the resulting message  $m$  to Bob, who uses  $m$  together with  $y$  to verify whether or not  $f(x, y) = 1$ . The communication complexity in this case is the minimum  $|m|$  over all protocols. Such a protocol is *unambiguous* iff for each  $(x, y)$  pair such that  $f(x, y) = 1$  there is precisely one message  $m$  that will cause Bob to output 1. A set of the form  $A \times B$  for  $A \subseteq \{0, 1\}^{\mathbf{X}}$ ,  $B \subseteq \{0, 1\}^{\mathbf{Y}}$  is called a *rectangle*. The minimum of  $|m|$  over all unambiguous protocols is the *unambiguous communication complexity* of  $f$ ; it is known to be the logarithm base 2 of the minimum number of rectangles into which one can partition the set of inputs on which  $f$  is 1.

A canonical hard problem for communication complexity is the two-party disjointness (set intersection) problem,

$\bigvee_{i=1}^n x_i \wedge y_i$  where  $x$  and  $y$  are indicator vectors of sets in  $[n]$ . It has deterministic communication complexity  $n + 1$  (and requires  $\Omega(n)$  bits be sent even with randomness, but that is beyond what we need). We will need a variant of the “best partition” version of communication complexity in which the protocol includes a choice of the best split of input indices  $\mathbf{X}$  and  $\mathbf{Y}$  between Alice and Bob.

A typical method for proving lower bounds on OBDD size for a Boolean function  $f$  begins by observing that a size  $s$  OBDD may be simulated by a  $\log s$ -bit one-way communication protocol where Alice holds the first half of the variables read by the OBDD and Bob holds the second half. In this protocol, Alice starts at the root of the OBDD and follows the (unique) OBDD path determined by her half of the input until she reaches a node  $v$  querying a variable held by Bob. She then sends the identity of the node  $v$  to Bob, who can finish the computation starting from  $v$ . Thus, if we show that  $f$  has one-way communication complexity  $CC_{\mathbf{X} \rightarrow \mathbf{Y}}(f(\mathbf{X}, \mathbf{Y}))$  at least  $C$  in the best split  $\{\mathbf{X}, \mathbf{Y}\}$  of its input variables, then any OBDD computing  $f$  must have at least  $2^C$  nodes.

Our lower bound for SDDs uses related ideas but in a more sophisticated way, and instead of providing a one-way deterministic protocol, we give an unambiguous protocol that simulates the SDD computation. In particular, the conversion to deterministic protocols requires two-way communication.

### 3 SDDs and Best-Partition Communication Complexity

In this section, we show how we can use any small SDD representing a function  $f$  to build an efficient communication protocol for  $f$  given an approximately balanced partition of input variables that is determined by its associated vtree. As a consequence, any function requiring large communication complexity under all such partitions requires large SDDs. To begin this analysis, we consider how an SDD simplifies under a partial assignment to its input variables.

#### 3.1 Pruning SDDs Using Restrictions

**Definition 3.1.** Suppose that  $\mathbf{v}$  is a pruned vtree for a set of variables  $\mathbf{X}$ , and that  $v$  is a vertex in  $\mathbf{v}$ . Let  $\text{Vars}(v)$  denote the set of variables that are descendants of  $v$  in  $\mathbf{v}$  and  $\text{Shell}(v) = \mathbf{X} \setminus \text{Vars}(v)$ . Also let  $\text{Parent}(v)$  denote the (unique) vertex in  $\mathbf{v}$  that has  $v$  as a child.

We define a construction to capture what happens to an SDD under a partial assignment of its variables.

**Definition 3.2.** Let  $\alpha$  be an SDD that respects  $\mathbf{v}$ , a vtree for the variables  $\mathbf{X}$ , and suppose that  $\alpha$  computes the function  $f$ . Let  $\mathbf{B} \subseteq \mathbf{X}$  and  $\mathbf{A} = \mathbf{X} \setminus \mathbf{B}$  and let  $\rho : \mathbf{A} \rightarrow \{0, 1\}$  be

an assignment to the variables in  $\mathbf{A}$ . Let  $\alpha|_{\rho}$  be Boolean circuit remaining after plugging the partial assignment  $\rho$  into the SDD  $\alpha$  and making the following simplifications:

1. If a gate computes a constant  $c \in \{\top, \perp\}$  under the partial assignment  $\rho$ , we can replace that gate and its outgoing edges with  $c$ .
2. Remove any children of OR-gates that compute  $\perp$ .
3. Remove any nodes disconnected from the root.

For each vtree vertex  $v \in \mathbf{v}$  that was not removed in this process, we denote its counterpart in the pruned vtree  $\mathbf{v}|_{\mathbf{A}}$  by  $v|_{\mathbf{A}}$ .

Construct the pruned vtree  $\mathbf{v}|_{\mathbf{A}}$  from  $\mathbf{v}$  as follows: for each vertex  $v$ , if  $\text{Vars}(v) \subseteq \mathbf{A}$  and  $\text{Vars}(\text{Parent}(v)) \not\subseteq \mathbf{A}$ , replace  $v$  and its subtree by a stub. We say that we have pruned the subtree rooted at  $v$ .

For  $\mathbf{A} \subseteq \mathbf{X}$ , we call  $\{\mathbf{A}, \mathbf{X} \setminus \mathbf{A}\}$  a shell partition for  $\mathbf{X}$  if there is a vtree vertex  $v \in \mathbf{v}$  such that  $\text{Shell}(v) = \mathbf{A}$ . We call  $\mathbf{A}$  the shell. If, for a restriction  $\rho : \mathbf{A} \rightarrow \{0, 1\}$ , there exists a vtree vertex  $v \in \mathbf{v}$  such that  $\text{Shell}(v) = \mathbf{A}$ , we call  $\rho$  a shell restriction.

**Proposition 3.3.** Let  $\alpha$  be an SDD that respects  $\mathbf{v}$ , a vtree for the variables  $\mathbf{X}$ , and suppose that  $\alpha$  computes the function  $f$ . Let  $\mathbf{A} \subseteq \mathbf{X}$  and  $\rho : \mathbf{A} \rightarrow \{0, 1\}$  be a partial assignment of the variables in  $\mathbf{A}$ . The pruned SDD  $\alpha|_{\rho}$  has the following properties:

- (a)  $\langle \alpha|_{\rho} \rangle = f|_{\rho}$ .
- (b)  $\alpha|_{\rho}$  is a pruned SDD respecting  $\mathbf{v}|_{\mathbf{A}}$ .
- (c)  $G(\alpha|_{\rho})$  is a subgraph of  $G(\alpha)$ .

*Proof.* (a): An SDD may be equivalently described as a Boolean circuit of alternating OR and AND gates. For any Boolean circuit in the variables  $\mathbf{X}$  that computes  $f$ , plugging in the values for the restriction  $\rho$  yields a circuit com-

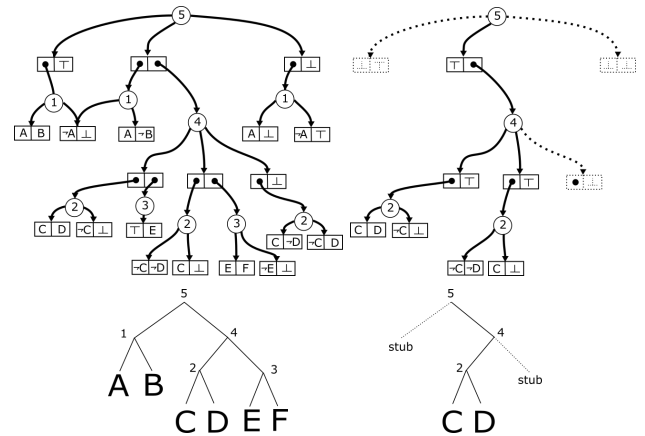


Figure 2: An SDD and its vtree, as well as the pruned pair after setting  $B$  to 0 and  $A, E, F$  to 1.



puting  $f|_\rho$ . Furthermore, the simplification steps do not change the function computed.

(b): For each  $v$  such that  $\text{Vars}(v) \subseteq \mathbf{A}$  and  $\text{Vars}(\text{Parent}(v)) \not\subseteq \mathbf{A}$ , we have replaced the subtree rooted at  $v$  by a stub and replaced the SDDs in  $\alpha$  respecting  $v$  by either  $\top$  or  $\perp$ . Thus  $\alpha|_\rho$  respects  $\mathbf{v}|_\mathbf{A}$ .

We now check that  $\alpha|_\rho$  is a pruned SDD. In particular we need to ensure that for each SDD  $\alpha' = \{(p_1, s_1), \dots, (p_\ell, s_\ell)\}$  in  $\alpha$ , the corresponding pruned SDDs that remain from  $p_1, \dots, p_\ell$  in its pruned counterpart  $\alpha'|_\rho$  represent a collection of disjoint functions. From the first part of this proposition, these are  $\langle p_{i_1} \rangle|_\rho, \dots, \langle p_{i_k} \rangle|_\rho$  for some  $k \leq n$ , where we have only included those SDDs that are consistent under  $\rho$ . Since the original set of SDDs was a partition and thus disjoint, this set of restricted (pruned) SDDs is also disjoint.

(c): The process in Definition 3.2 only removes nodes from  $G(\alpha)$  to construct  $G(\alpha|_\rho)$ . Further, it does not change the label of any SDD that was not removed.  $\square$

### 3.2 Unambiguous Communication Protocol for SDDs

The way that we will partition the input variables to an SDD between the parties Alice and Bob in the communication protocol will respect the structure of its associated vtree. The restrictions will correspond to assignments that reflect Alice's knowledge of the input and will similarly respect that structure.

Notice that a vtree cut along an edge  $(u, v)$  (where  $u$  is the parent of  $v$ ) induces a shell partition for  $\mathbf{X}$  consisting of the set  $\mathbf{B} = \text{Vars}(v)$ , and the shell  $\mathbf{A} = \mathbf{X} \setminus \mathbf{B}$ .

**Proposition 3.4.** *Let  $\alpha$  be an SDD of size  $s$  computing a function  $f : \{0, 1\}^{\mathbf{X}} \rightarrow \{0, 1\}$  that respects a vtree  $\mathbf{v}$ . Suppose that  $\{\mathbf{A}, \mathbf{B}\}$  is a shell partition for  $\mathbf{X}$  and that  $\mathbf{A}$  is its shell. Let  $b$  be the vertex in  $\mathbf{v}$  for which  $\text{Vars}(b) = \mathbf{B}$  and  $\text{Vars}(\text{Parent}(b)) \not\subseteq \mathbf{B}$ .*

*For any shell restriction  $\rho : \mathbf{A} \rightarrow \{0, 1\}$ , the set  $\langle \text{Sdds}_{\alpha|_\rho}(b|_\mathbf{A}) \rangle$  is a disjoint collection of functions.*

*Proof.* For non-shell restrictions  $\rho'$ , the collection of functions  $\langle \text{Sdds}_{\alpha|_{\rho'}}(v) \rangle$  for a vtree node  $v$  is not disjoint; we need to use the specific properties of  $\mathbf{A}$  and  $b$ . Since  $\rho$  was a shell restriction, the pruned vtree  $\mathbf{v}|_\mathbf{A}$  takes the form of a path  $v_1|_\mathbf{A}, \dots, v_k|_\mathbf{A}$  of internal vertices, where  $v_1$  is the root of  $\mathbf{v}$ , and  $v_k|_\mathbf{A} = b|_\mathbf{A}$ , with the other child of each of  $v_1|_\mathbf{A}, \dots, v_{k-1}|_\mathbf{A}$  being a stub, together with a vtree for the variables  $\mathbf{B}$  rooted at  $b$ . We will show that if  $\langle \text{Sdds}_{\alpha|_\rho}(v_i|_\mathbf{A}) \rangle$  is disjoint then so is  $\langle \text{Sdds}_{\alpha|_\rho}(v_{i+1}|_\mathbf{A}) \rangle$ . This will prove the proposition since  $\langle \text{Sdds}_{\alpha|_\rho}(v_1|_\mathbf{A}) \rangle$  only contains the function  $\langle \alpha|_\rho \rangle$  and is therefore trivially disjoint.

We will use the fact that every pruned-SDD from

$\text{Sdds}_{\alpha|_\rho}(v_{i+1}|_\mathbf{A})$  is contained in some SDD from  $\text{Sdds}_{\alpha|_\rho}(v_i|_\mathbf{A})$ . We have two cases to check:  $v_{i+1}|_\mathbf{A}$  is either a left child or a right child of  $v_i|_\mathbf{A}$ .

If  $v_{i+1}|_\mathbf{A}$  was a right child then each pruned-SDD  $\eta|_\rho$  contained in  $\text{Sdds}_{\alpha|_\rho}(v_i|_\mathbf{A})$  takes the form  $\eta|_\rho = \{(\top, s|_\rho)\}$ . Then  $\langle \text{Sdds}_{\alpha|_\rho}(v_{i+1}|_\mathbf{A}) \rangle = \langle \text{Sdds}_{\alpha|_\rho}(v_i|_\mathbf{A}) \rangle$  and is therefore disjoint by assumption.

Otherwise suppose that  $v_{i+1}|_\mathbf{A}$  is the left child of  $v_i|_\mathbf{A}$ . Let  $\eta|_\rho \in \text{Sdds}_{\alpha|_\rho}(v_i|_\mathbf{A})$ . Let  $\eta|_\rho = \{(\eta_1|_\rho, \top), \dots, (\eta_k|_\rho, \top)\}$  where  $\bigvee_{i=1}^k \langle \eta_i|_\rho \rangle = \langle \eta|_\rho \rangle$  and  $\{\langle \eta_1|_\rho \rangle, \dots, \langle \eta_k|_\rho \rangle\}$ , being a collection of primes for  $\eta|_\rho$ , is disjoint. By assumption  $\langle \text{Sdds}_{\alpha|_\rho}(v_i|_\mathbf{A}) \rangle$  is disjoint, so for any other  $\eta'|_\rho = \{(\eta'_1|_\rho, \top), \dots, (\eta'_{k'}|_\rho, \top)\} \in \text{Sdds}_{\alpha|_\rho}(v_i|_\mathbf{A})$  distinct from  $\eta|_\rho$ , we have  $\langle \eta|_\rho \rangle \wedge \langle \eta'|_\rho \rangle = \perp$ . Then for any  $i \in [k]$  and  $j \in [k']$ , we have  $\langle \eta_i|_\rho \rangle \wedge \langle \eta'_j|_\rho \rangle = \perp$ . Thus  $\langle \text{Sdds}_{\alpha|_\rho}(v_{i+1}|_\mathbf{A}) \rangle$  is disjoint.  $\square$

**Theorem 3.5.** *Let  $\alpha$  be an SDD of size  $s$  that respects a vtree  $\mathbf{v}$  and suppose that it computes the function  $f : \{0, 1\}^{\mathbf{X}} \rightarrow \{0, 1\}$ . Suppose that  $\{\mathbf{A}, \mathbf{B}\}$  is a shell partition for  $\mathbf{X}$  and that  $\mathbf{A}$  is the shell. Let  $b$  be the vertex in  $\mathbf{v}$  for which  $\text{Vars}(b) = \mathbf{B}$  and  $\text{Vars}(\text{Parent}(b)) \not\subseteq \mathbf{B}$ .*

*Consider the communication game where Alice has the variables  $\mathbf{A}$ , Bob has the variables  $\mathbf{B}$ , and they are trying to compute  $f(\mathbf{A}, \mathbf{B})$ . There is a  $\log s$ -bit unambiguous communication protocol computing  $f$ .*

*Proof.* Suppose that Alice and Bob both know the SDD  $\alpha$ . Let  $\rho : \mathbf{A} \rightarrow \{0, 1\}$  be the partial assignment corresponding to Alice's input. This is a shell restriction. Alice may then privately construct the pruned SDD  $\alpha|_\rho$ , which computes  $f|_\rho$  by Proposition 3.3. Further,  $\alpha|_\rho$  evaluates to 1 under Bob's input  $\phi : \mathbf{B} \rightarrow \{0, 1\}$  if and only if there exists a pruned-SDD  $\eta|_\rho \in \text{Sdds}_{\alpha|_\rho}(b|_\mathbf{A})$  such that  $\langle \eta|_\rho \rangle(\phi) = 1$ .

By Proposition 3.4,  $\langle \text{Sdds}_{\alpha|_\rho}(b|_\mathbf{A}) \rangle$  is disjoint. Also, since  $\rho$  is a shell restriction with shell  $\mathbf{A}$ , and  $\text{Vars}(b) = \mathbf{B} = \mathbf{X} \setminus \mathbf{A}$ , every SDD in  $\text{Sdds}_{\alpha|_\rho}(b|_\mathbf{A})$  was unchanged by  $\rho$ . In particular, this means that  $\text{Sdds}_{\alpha|_\rho}(b|_\mathbf{A}) \subseteq \text{Sdds}_\alpha(b)$  and any pruned-SDD  $\eta|_\rho$  can be viewed as some  $\eta \in \text{Sdds}_\alpha(b)$  that is also in  $\text{Sdds}_{\alpha|_\rho}(b|_\mathbf{A})$ .

For the protocol Alice nondeterministically selects an  $\eta$  from  $\text{Sdds}_{\alpha|_\rho}(b|_\mathbf{A})$  and then sends its identity as a member of  $\text{Sdds}_\alpha(b)$  to Bob. This requires at most  $\log s$  bits. Bob will output 1 on his input  $\phi$  if and only if  $\langle \eta \rangle(\phi) = 1$ , which he can test since he knows  $\alpha$  and  $b$ . This protocol is unambiguous since the fact that  $\langle \text{Sdds}_{\alpha|_\rho}(b|_\mathbf{A}) \rangle$  is disjoint means that for any input  $\phi$  to Bob there is at most one  $\eta \in \text{Sdds}_{\alpha|_\rho}(b|_\mathbf{A})$  such that  $\langle \eta \rangle(\phi) = 1$ . Since Bob knows  $\alpha$ , he also knows  $\eta$  and can therefore compute  $\langle \eta \rangle(\phi)$ . Since  $\alpha$  computes  $f$ , if  $\langle \eta \rangle(\phi) = 1$  then  $f(\phi, \rho) = 1$ . Otherwise all of the functions in  $\langle \text{Sdds}_{\alpha|_\rho}(b|_\mathbf{A}) \rangle$  evaluate to 0 on input  $\phi$  so  $f(\phi, \rho) = 0$ .  $\square$

We can relate the deterministic and unambiguous communication complexities of a function using the following result from [Yannakakis, 1991].

**Theorem 3.6** (Yannakakis). *If there is an  $g$ -bit unambiguous communication protocol for a function  $f : \{0, 1\}^{\mathbf{A}} \times \{0, 1\}^{\mathbf{B}} \rightarrow \{0, 1\}$ , then there is a  $(g+1)^2$ -bit deterministic protocol for  $f$ .*

The following  $1/3$ - $2/3$  lemma is standard.

**Lemma 3.7.** *For a vtree  $\mathbf{v}$  for  $L$  variables, if a vertex  $b$  satisfies  $\frac{1}{3}L \leq |\text{Vars}(b)| \leq \frac{2}{3}L$ , we call it a  $(1/3, 2/3)$  vertex. Every vtree contains a  $(1/3, 2/3)$  vertex.*

**Definition 3.8.** *Let  $\mathbf{X}$  be a set of variables and  $(\mathbf{A}, \mathbf{B})$  a partition of  $\mathbf{X}$ . We call the partition  $(\mathbf{A}, \mathbf{B})$  a  $(\delta, 1 - \delta)$ -partition for  $\delta \in [0, 1/2]$  if  $\min(|\mathbf{A}|, |\mathbf{B}|) \geq \delta|\mathbf{X}|$ . That is, the minimum size of one side of the partition is at least a  $\delta$ -fraction of the total number of variables.*

*The best  $(\delta, 1 - \delta)$ -partition communication complexity of a Boolean function  $f : \{0, 1\}^{\mathbf{X}} \rightarrow \{0, 1\}$  is  $\min(CC(f(\mathbf{A}, \mathbf{B})))$  where the minimum is taken over all  $(\delta, 1 - \delta)$ -partitions  $(\mathbf{A}, \mathbf{B})$ .*

**Theorem 3.9.** *If the best  $(1/3, 2/3)$ -partition communication complexity of a Boolean function  $f : \{0, 1\}^{\mathbf{X}} \rightarrow \{0, 1\}$  is  $C$ , then an SDD computing  $f$  has size at least  $2^{\sqrt{C}-1}$ .*

*Proof.* Suppose that  $\alpha$  is an SDD of size  $s$  respecting the vtree  $\mathbf{v}$  for variables  $\mathbf{X}$ , and that  $\alpha$  computes  $f$ . From Lemma 3.7 the vtree  $\mathbf{v}$  contains a  $(1/3, 2/3)$  vertex  $b$ . This  $(1/3, 2/3)$  vertex  $b$  induces a  $(1/3, 2/3)$ -partition of the variables  $\{\mathbf{A}, \mathbf{B}\}$  where  $\mathbf{B} = \text{Vars}(b)$  and  $\mathbf{A} = \text{Shell}(b)$ . Further, this partition  $\{\mathbf{A}, \mathbf{B}\}$  is a shell partition. By Theorem 3.5, there exists a  $\log s$ -bit unambiguous communication protocol for  $f(\mathbf{A}, \mathbf{B})$ . Then by Theorem 3.6, there exists a  $(\log(s) + 1)^2$ -bit deterministic communication protocol for  $f(\mathbf{A}, \mathbf{B})$ . Since the best  $(1/3, 2/3)$ -partition communication complexity of  $f$  is  $C$ , we have that  $C \leq (\log(s) + 1)^2$  which implies that  $s \geq 2^{\sqrt{C}-1}$  as stated.  $\square$

## 4 Lower Bounds for SDDs

There are a large number of predicates  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  for which the  $(1/3, 2/3)$ -partition communication complexity is  $\Omega(n)$  and by Theorem 3.9 each of these requires SDD size  $2^{\Omega(\sqrt{n})}$ . The usual best-partition communication complexity is  $(1/2, 1/2)$ -partition communication complexity. For example, the function SHIFTEDEQ which takes as inputs  $x, y \in \{0, 1\}^n$  and  $z \in \{0, 1\}^{\lceil \log_2 n \rceil}$  and tests whether or not  $y = \text{SHIFT}(x, z)$  where  $\text{SHIFT}(x, z)$  is the cyclic shift of  $x$  by  $(z)_2$  positions. However, as is typical of these functions, the same proof which shows that the  $(1/2, 1/2)$ -partition communication complexity of SHIFTEDEQ is  $\Omega(n)$  also shows

that its  $(1/3, 2/3)$ -partition communication complexity is  $\Omega(n)$ . However, most of these functions are not typical of predicates to which one might want to apply weighted model counting. Instead we analyze SDDs for formulas derived from a natural class of database queries. We are able to characterize SDD size for these queries, proving exponential lower bounds for every such query that cannot already be represented in linear size by an OBDD. This includes an example of a query called  $Q_V$  for which FBDDs are polynomial size but the best SDD requires exponential size.

### 4.1 SDD Knowledge Compilation for Database Query Lineages

We analyze SDDs for a natural class of database queries called the *union of conjunctive queries (UCQ)*. This includes all queries given by the grammar

$$q ::= R(\mathbf{x}) \mid \exists x q \mid q \wedge q \mid q \vee q$$

where  $R(\mathbf{x})$  is an elementary relation and  $x$  is a variable. For each such query  $q$ , given an input database  $D$ , the query's *lineage*,  $\Phi_q^D$ , is a Boolean expression for  $q$  over Boolean variables that correspond to tuples in  $D$ . In general, one thinks of the query size as fixed and considers the complexity of query evaluation as a function of the size of the database. The following formulas are lineages (or parts thereof) of well-known queries that are fundamental for probabilistic databases [Dalvi and Suciu, 2012, Jha and Suciu, 2013] over a particular database  $D_0$  (called the *complete bipartite graph of size  $m$*  in [Jha and Suciu, 2013]):

$$H_0 = \bigvee_{i,j \in [m]} R_i S_{ij} T_j$$

$$Q_V = \bigvee_{i,j \in [m]} R_i S_{ij} \vee S_{ij} T_j \vee R_i T_j$$

$$H_1 = \bigvee_{i,j \in [m]} R_i S_{ij} \vee S_{ij} T_j$$

$$H_{k0} = \bigvee_{i \in [m]} R_i S_{ij}^1 \quad \text{for } k \geq 1$$

$$H_{k\ell} = \bigvee_{i,j \in [m]} S_{ij}^\ell S_{ij}^{\ell+1} \quad \text{for } 0 < \ell < k$$

$$H_{kk} = \bigvee_{i \in [m]} S_{ij}^k T_j \quad \text{for } k \geq 1.$$

(The corresponding queries are represented using lower case letters  $h_0, q_V, h_1, h_{k0}, \dots, h_{kk}$  and involve unary relations  $R$  and  $T$ , as well as binary relations  $S$  and  $S^k$ . For example,  $h_0 = \exists x_0 \exists y_0 R(x_0) S(x_0, y_0) T(y_0)$ .) The following lemma will be useful in identifying subformulas of the above query lineages that can be used to compute the set disjointness function.

**Proposition 4.1.** *Let the elements of  $[m] \times [m]$  be partitioned into two sets  $A$  and  $B$ , each of size at least  $\delta m^2$ . Let  $\text{Row}(i)$  denote  $\{i\} \times [m]$  and  $\text{Col}(j)$  denote  $[m] \times \{j\}$ . Define  $W_{\text{Row}} = \{i \in [m] \mid \emptyset \neq \text{Row}(i) \cap A \text{ and } \emptyset \neq \text{Row}(i) \cap B\}$ . That is,  $\text{Row}(i)$  for  $i \in W_{\text{Row}}$  is split into two nonempty pieces by the partition. Similarly, define  $W_{\text{Col}} = \{j \in [m] \mid \emptyset \neq \text{Col}(j) \cap A \text{ and } \emptyset \neq \text{Col}(j) \cap B\}$ . Then*

$$\max(|W_{\text{Row}}|, |W_{\text{Col}}|) \geq \sqrt{\delta} \cdot m.$$

*Proof.* Suppose that both  $|W_{\text{Row}}| < m$  and  $|W_{\text{Col}}| < m$ . By definition, if  $i \notin W_{\text{Row}}$  then one of  $A$  or  $B$  contains an entire row,  $\text{Row}(i)$ , say  $A$  without loss of generality. This implies that no column  $\text{Col}(j)$  is entirely contained in  $B$ . Since  $|W_{\text{Col}}| < m$ , there is some column  $\text{Col}(j)$  that is entirely contained in  $A$ . This in turn implies that  $B$  does not contain any full row. In particular, we have that  $A$  contains all rows in  $[m] \setminus W_{\text{Row}}$  and all columns in  $[m] \setminus W_{\text{Col}}$  and thus  $B \subseteq W_{\text{Row}} \times W_{\text{Col}}$  and so  $|B| \leq |W_{\text{Row}}| \cdot |W_{\text{Col}}|$ . By assumption,  $|B| \geq \delta m^2$ . Hence  $|W_{\text{Row}}| \cdot |W_{\text{Col}}| \geq \delta m^2$  and so  $\max\{|W_{\text{Row}}|, |W_{\text{Col}}|\} \geq \sqrt{\delta} \cdot m$ .  $\square$

**Theorem 4.2.** *For  $m \geq 6$ , the best  $(1/3, 2/3)$ -partition communication complexity of  $Q_V$ ,  $H_0$ , and of  $H_1$  is at least  $m/3$ .*

*Proof.* Let  $\mathbf{X}$  be the set of variables appearing in  $Q_V$  (or  $H_1$ ) and let  $(\mathbf{A}, \mathbf{B})$  be a  $(1/3, 2/3)$ -partition of  $\mathbf{X}$ . Let  $(A, B)$  be the partition of  $[m] \times [m]$  induced by  $(\mathbf{A}, \mathbf{B})$  and define  $W_{\text{Row}}$  and  $W_{\text{Col}}$  as in Proposition 4.1. Since  $|\mathbf{X}| = m^2 + 2m$  and only elements of  $[m] \times [m]$  are relevant,  $|A|, |B| \geq (m^2 + 2m)/3 - 2m = (1 - 4/m)m^2/3 \geq m^2/9$  for  $m \geq 6$  and hence  $\max(|W_{\text{Row}}|, |W_{\text{Col}}|) \geq m/3$ . We complete the proof by showing that computing  $Q_V(\mathbf{A}, \mathbf{B})$  and  $H_1(\mathbf{A}, \mathbf{B})$  each require at least  $\max(|W_{\text{Row}}|, |W_{\text{Col}}|)$  bits of communication between Alice and Bob. We will do this by showing that for a particular subset of inputs,  $Q_V$  is equivalent to the disjointness function for a  $\max(|W_{\text{Row}}|, |W_{\text{Col}}|)$  size set.

Suppose without loss of generality that  $|W_{\text{Row}}| \geq |W_{\text{Col}}|$ . Set all  $T_j = 0$  and for each  $i \notin W_{\text{Row}}$  set  $R_i = 0$ . For each  $i \in W_{\text{Row}}$  for which  $R_i \in \mathbf{A}$ , set all  $S_{ij} \in \mathbf{A}$  to 0, let  $j_i$  be minimal such that  $S_{ij_i} \in \mathbf{B}$ , and set  $S_{ij} \in \mathbf{B}$  to 0 for all  $j > j_i$ . (Such an index  $j_i$  must exist since  $i \in W_{\text{Row}}$ .) Similarly, For each  $i \in W_{\text{Row}}$  for which  $R_i \in \mathbf{B}$ , set all  $S_{ij} \in \mathbf{B}$  to 0, let  $j_i$  be minimal such that  $S_{ij_i} \in \mathbf{A}$ , and set  $S_{ij} \in \mathbf{A}$  to 0 for all  $j > j_i$ . In particular, under this partial assignment, we have

$$Q_V = H_1 = \bigvee_{i \in W_{\text{Row}}} R_i S_{ij_i}$$

and for each  $i \in W_{\text{Row}}$ , Alice holds one of  $R_i$  or  $S_{ij_i}$  and Bob holds the other. We can reduce  $H_0$  to the same quantity by setting all  $T_j = 1$ . This is precisely the set disjointness problem on two sets of size  $|W_{\text{Row}}|$  where membership of  $i$  in each player's set is determined by the value of the unset bit indexed by  $i$  that player holds. Therefore, computing

$Q_V$  or  $H_1$  requires at least  $|W_{\text{Row}}|$  bits of communication, as desired.  $\square$

Combining this with Theorem 3.9, we immediately obtain the following:

**Theorem 4.3.** *For  $m \geq 6$ , any SDD representing  $Q_V$  or  $H_1$  requires size at least  $2\sqrt{m/3-1}$ .*

As [Jha and Suciu, 2013] has shown that  $Q_V$  has FBDD size  $O(m^2)$ , we obtain the following separation.

**Corollary 4.4.** *FBDDs can be exponentially more succinct than SDDs. In particular,  $Q_V$  has FBDD size  $O(m^2)$  but every SDD for  $Q_V$  requires size  $2\sqrt{m/3-1}$  for  $m \geq 6$ .*

We now consider the formulas  $H_{ki}$  above. Though they seem somewhat specialized, these formulas are fundamental to UCQ queries: [Jha and Suciu, 2013] define the notion of an *inversion* in a UCQ query and use it to characterize the OBDD size of UCQ queries. In particular they show that if a query  $q$  is *inversion-free* then the OBDD size of its lineage  $Q$  is linear and if  $q$  has an minimum inversion length  $k \geq 1$  then it requires OBDD size  $2^{\Omega(n/k)}$  where  $n$  is the domain size of all attributes. Jha and Suciu obtain this lower bound by analyzing the  $H_{ki}$  we defined above. (We will not define the notion of inversions, or their lengths, and instead use the definition as a black box. However, as an example, the query associated with  $H_1$  has an inversion of length 1 so its OBDD size is  $2^{\Omega(m)}$ .)

**Proposition 4.5.** [Jha and Suciu, 2013] *Let  $q$  be a query with a length  $k \geq 1$  inversion. Let  $D_0$  be the complete bipartite graph of size  $m$ . There exists a database  $D$  for  $q$ , along with variable restrictions  $\rho_i$  for all  $i \in [0, k]$ , such that  $|D| = O(|D_0|)$  and  $\Phi_q^D|_{\rho_i} = \Phi_{h_{ki}}^{D_0} = H_{ki}$*

**Theorem 4.6.** *Let  $k \geq 2$  and assume that  $m \geq 6$ . Let  $q$  be a query with a length  $k \geq 2$  inversion. Then there exists a database  $D$  for which any SDD for  $Q = \Phi_q^D$  has size at least  $2\sqrt{m/k/3-1}$ .*

*Proof.* Given a query  $q$ , let  $D$  be the database for  $q$  constructed in Proposition 4.5. Fix the vtree  $\mathbf{v}$  over  $\mathbf{X}_k$  respected by an SDD  $\alpha$  for  $\Phi_q^D$ . By Lemma 3.7, there exists a  $(1/3, 2/3)$  node  $b$  in the vtree  $\mathbf{v}$  that gives a  $(1/3, 2/3)$  partition  $\{\mathbf{A}, \mathbf{B}\}$  of  $\mathbf{X}_k$ . By Proposition 4.5, there are restrictions  $\rho_0, \dots, \rho_k$  such that  $\Phi_q^D|_{\rho_i} = H_{ki}$  for all  $i$ . Thus  $\alpha|_{\rho_i}$  is a (pruned) SDD, of size  $\leq$  that of  $\alpha$ , respecting  $\mathbf{v}|_{\rho_i}$  and computing  $H_{ki}$ . Observe that the restriction of  $\{\mathbf{A}, \mathbf{B}\}$  to the variables of  $\mathbf{X}_{ki}$  is also shell partition of  $\mathbf{v}|_{\rho_i}$  at node  $b$ .

We will show that there must exist an  $H_{ki}$  for which  $\text{CC}(H_{ki}(\mathbf{A}, \mathbf{B})) \geq m/(9k)$  and therefore by Theorem 3.6, this implies that the unambiguous communication complexity of  $H_{ki}$  is at least  $\frac{1}{3}\sqrt{m/k-1}$ . Then by Theorem 3.5, any SDD respecting  $\mathbf{v}$  that computes  $H_{ki}$  has size at least  $2^{\frac{1}{3}\sqrt{m/k-1}}$ .

Let  $W_{\text{Chain}}$  contain all pairs  $(i, j)$  for which both  $\mathbf{A} \cap \bigcup_{\ell=1}^k \{S_{ij}^\ell\} \neq \emptyset$  and  $\mathbf{B} \cap \bigcup_{\ell=1}^k \{S_{ij}^\ell\} \neq \emptyset$  and Let  $\gamma = 1/9$ . We will consider two cases: either  $|W_{\text{Chain}}| \geq \gamma \cdot m$  or  $|W_{\text{Chain}}| < \gamma \cdot m$ .

In the first case, since  $|W_{\text{Chain}}| \geq \gamma \cdot m$ , there must exist at least  $\gamma \cdot m$  tuples  $(i, j, \ell)$  for which either  $S_{ij}^\ell \in \mathbf{A}$  and  $S_{ij}^{\ell+1} \in \mathbf{B}$  or vice-versa. Call the set of these tuples  $\mathbf{T}$ . Then, since there are  $k - 1$  choices of  $\ell < k$ , there exists some  $\ell^*$  such that the set  $\mathbf{T}_{\ell^*} := \mathbf{T} \cap [m] \times [m] \times \{\ell^*\}$  contains at least  $\gamma \cdot m / (k - 1) > m / (9k)$  elements. If we set all variables of  $\mathbf{X}_{k\ell^*}$  outside of  $\mathbf{T}_{\ell^*}$  to 0, the function  $H_{k\ell^*}$  corresponds to solving a disjointness problem between Alice and Bob on the elements of  $\mathbf{T}_{\ell^*}$ . Thus the communication complexity of  $H_{k\ell^*}$  under the partition  $\{\mathbf{A}, \mathbf{B}\}$  is at least  $m / (9k)$ .

In the second case, consider the largest square submatrix  $M$  of  $[m] \times [m]$  that does not contain any member of  $W_{\text{Chain}}$ . We mimic the argument of Theorem 4.2 on this submatrix  $M$ . By definition,  $M$  has side  $m' \geq (1 - \gamma)m$ . For every  $(i, j)$  in  $M$ , either  $\mathbf{A}$  or  $\mathbf{B}$  contains all  $S_{ij}^\ell$ ; let  $A$  be those  $(i, j)$  such that these are in  $\mathbf{A}$  and  $B$  be those  $(i, j)$  for which they are in  $\mathbf{B}$ . Since  $|\mathbf{A}|, |\mathbf{B}| \geq |\mathbf{X}_k|/3 = (km^2 + 2m)/3$  and there are at most  $2m + (\gamma^2 + 2\gamma)km^2$  variables not in  $M$ ,

$$\begin{aligned} |A|, |B| &\geq [(km^2 + 2m)/3 - 2m + (\gamma^2 + 2\gamma)km^2]/k \\ &= [(1 - \gamma)^2 - 2/3 - 4/(3km)]m^2 > (m/18)^2 \end{aligned}$$

since  $k \geq 2$ . Applying Proposition 4.1, we see that  $\max(|W_{\text{Row}}|, |W_{\text{Col}}|) \geq m/18 \geq m/(9k)$ . By the same argument presented in the proof of Theorem 4.2, we have both  $\text{CC}(H_{k0}(\mathbf{A}, \mathbf{B})) \geq |W_{\text{Row}}|$  and  $\text{CC}(H_{kk}(\mathbf{A}, \mathbf{B})) \geq |W_{\text{Col}}|$  so at least one of these is at least  $m/(9k)$  and the theorem follows.  $\square$

It follows that for inversion-free UCQ queries, both SDD and OBDD sizes of any lineage are linear, while UCQ queries with inversions (of length  $k$ ) have worst-case lineage size that is exponential ( $2^{\Omega(m/k)}$  for OBDDs and  $2^{\Omega(\sqrt{m/k})}$  for SDDs). Note that the same SDD size lower bound for UCQ query lineage  $Q = \Phi_q^D$  applies to its dual  $Q^* = \Phi_q^D$  as follows: Flipping the signs on the variables in  $Q^*$  yields a function equivalent to  $\neg Q$ . So flipping the variable signs at the leaves of an SDD for  $Q^*$  we obtain an SDD of the same size for  $\neg Q$  and hence a deterministic protocol that also can compute  $Q$ .

## 5 Simulating DNNFs by OR-FBDDs

In this section, we extend the simulation of decision-DNNFs by FBDDs from [Beame et al., 2013] to obtain a simulation of general DNNFs by OR-FBDDs with at most a quasipolynomial increase in size. This simulation yields lower bounds on DNNF size from OR-FBDD lower bounds.

**Definition 5.1.** For each AND node  $u$  in a DNNF  $\mathcal{D}$ , let  $M_u$  be the number of AND nodes in the subgraph  $D_u$ . We call  $u$ 's left child  $u_l$  and its right child  $u_r$ . We will assume  $M_{u_l} \leq M_{u_r}$  (otherwise we swap  $u_l$  and  $u_r$ ).

For each AND node  $u$ , we classify the edge  $(u, u_l)$  as a light edge and the edge  $(u, u_r)$  a heavy edge. We classify every other edge in  $\mathcal{D}$  as a neutral edge.

For a DNNF  $\mathcal{D}$  or an OR-FBDD  $\mathcal{F}$ , we denote the functions that  $\mathcal{D}$  and  $\mathcal{F}$  compute as  $\Phi_{\mathcal{D}}$  and  $\Phi_{\mathcal{F}}$ .

### Constructing the OR-FBDD

For a DNNF  $\mathcal{D}$ , we will treat a leaf labeled by the variable  $X$  as a decision node that points to a 0-sink node if  $X = 0$  and a 1-sink node if  $X = 1$ , and vice-versa for a leaf labeled by  $\neg X$ . We also assume that each AND node has just two children, which only affects the DNNF size by at most polynomially.

**Definition 5.2.** Fix a DNNF  $\mathcal{D}$ . For a node  $u$  in  $\mathcal{D}$  and a path  $P$  from the root to  $u$ , let  $S(P)$  be the set of light edges along  $P$  and  $S(u) = \{S(P) \mid P \text{ is a path from the root to } u\}$ .

We will construct an OR-FBDD  $\mathcal{F}$  that computes the same boolean function as  $\mathcal{D}$ . Its nodes are pairs  $(u, s)$  where  $u$  is a node in  $\mathcal{D}$  and the set of light edges  $s$  belongs to  $S(u)$ . Its root is  $(\text{root}(\mathcal{D}), \emptyset)$ . The edges in  $\mathcal{F}$  are of three types:

*Type 1:* For each light edge  $e = (u, v)$  in  $\mathcal{D}$  and  $s \in S(u)$ , add the edge  $((u, s), (v, s \cup \{e\}))$  to  $\mathcal{F}$ .

*Type 2:* For each neutral edge  $e = (u, v)$  in  $\mathcal{D}$  and  $s \in S(u)$ , add the edge  $((u, s), (v, s))$  to  $\mathcal{F}$ .

*Type 3:* For each heavy edge  $(u, v_r)$ , let  $e = (u, v_l)$  be its sibling light edge. For each  $s \in S(u)$  and 1-sink node  $w$  in  $D_{v_l}$ , add the edge  $((w, s \cup \{e\}), (v_r, s))$  to  $\mathcal{F}$ .

We label the nodes  $u' = (u, s)$  as follows: (1) if  $u$  is a decision node in  $\mathcal{D}$  for the variable  $X$  then  $u'$  is a decision node in  $\mathcal{F}$  testing the same variable  $X$ , (2) if  $u$  is an AND-node, then  $u'$  is a no-op node, (3) if  $u$  is an OR node it remains an OR node. (4) if  $u$  is a 0-sink node, then  $u'$  is a 0-sink node, (5) if  $u$  is a 1-sink node, then: if  $s = \emptyset$  then  $u'$  is a 1-sink node, otherwise it is a no-op node.

We show an example of this construction in Figure 3.

### Size and Correctness

**Lemma 5.3.** For the DNNF  $\mathcal{D}$  let  $L$  denote the maximum number of light edges from the root to a leaf,  $M$  the number of AND nodes and  $N$  the total number of nodes. Then  $\mathcal{F}$  has at most  $NM^L$  nodes. Further, this is  $N \cdot 2^{\log^2 N}$ .

*Proof.* The nodes in  $\mathcal{F}$  are labeled  $(u, s)$ . There are  $N$  possible nodes  $u$  and at most  $M^L$  choices for the set  $s$ , as

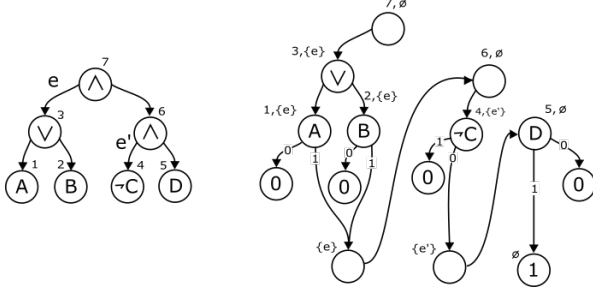


Figure 3: A DNNF and our construction of an equivalent OR-FBDD.

each path to  $u$  has at most  $L$  light edges.

Consider a root to leaf path with  $L$  light edges. As we traverse this path, every time we cross a light edge, we decrease the number of descendant AND nodes by more than half. Thus we must have begun with more than  $2^L$  descendant AND nodes at the root so that  $N \geq M > 2^L$ . This implies that  $NM^L$  is quasipolynomial in  $N$ ,

This upper bound is quasipolynomial in  $N$ , we will show that  $M > 2^L$ . Then, since  $N \geq M$ ,  $NM^L \leq N2^{\log^2 M} \leq N2^{\log^2 N}$ .  $\square$

The proof of the following lemma is in the full paper.

**Lemma 5.4.**  $\mathcal{F}$  is a correct OR-FBDD with no-op nodes that computes the same function as  $\mathcal{D}$ .

Using the quasipolynomial simulation of DNNFs by OR-FBDDs, we obtain DNNF lower bounds from OR-FBDD lower bounds.

**Definition 5.5.** Function  $\text{PERM}_n$  takes an  $n \times n$  boolean matrix  $M$  as input and outputs 1 if and only if  $M$  is a permutation matrix. The function  $\text{ROW-COL}_n$  takes an  $n \times n$  boolean matrix  $M$  as input and outputs 1 if and only if  $M$  has an all-0 row or an all-0 column.

**Theorem 5.6.** Any OR-FBDD computing  $\text{PERM}_n$  or  $\text{ROW-COL}_n$ , must have size  $2^{\Omega(n)}$  [Wegener, 2000].

**Corollary 5.7.** Any DNNF computing  $\text{PERM}_n$  or  $\text{ROW-COL}_n$  has size at least  $2^{\Omega(\sqrt{n})}$

## 6 Discussion

We have made the first significant progress in understanding the complexity of general DNNF representations. We have also provided a new connection between SDD representations and best-partition communication complexity. Best-partition communication complexity is a standard technique used to derive lower bounds on OBDD size, where it often yields asymptotically tight results. For communication lower bound  $C$ , the lower bound for OBDD size is  $2^C$  and the lower bound we have shown for SDD size is  $2^{\sqrt{C}} - 1$ . This is a quasipolynomial difference. Are SDDs that much more efficient than OBDDs? Is there always a

quasipolynomial simulation of SDDs by OBDDs in general, matching the quasipolynomial simulation of decision-DNNFs by FBDDs? Our separation result shows an example for which SDDs are sometimes exponentially less concise than FBDDs, and hence decision-DNNFs also. Are SDDs ever more concise than decision-DNNFs?

By plugging in the arguments of [Pipatsrisawat and Darwiche, 2010, Pipatsrisawat, 2010] in place of Theorem 3.5, all of our lower bounds immediately extend to size lower bounds for structured deterministic DNNFs (d-DNNFs), of which SDDs are a special case. It remains open whether structured d-DNNFs are strictly more concise than SDDs. [Pipatsrisawat and Darwiche, 2008, Pipatsrisawat, 2010] have proved an exponential separation between structured d-DNNFs and OBDDs using the *Indirect Storage Access (ISA)* function [Breitbart et al., 1995], but the small structured d-DNNF for this function is very far from an SDD. It is immediate that, under any variable partition, the  $ISA_n$  function has an  $O(\log n)$ -bit two-round deterministic communication protocol. On the other hand, efficient one-round (i.e., one-way) communication protocols yield small OBDDs so there are two possibilities if SDDs and structured d-DNNFs have different power. Either (1) communication complexity considerations on their own are not enough to derive a separation between SDDs and structured d-DNNFs, or (2) every SDD can be simulated by an efficient one-way communication protocol, in which case SDDs can be simulated efficiently by OBDDs (though the ordering cannot be the same as the natural traversal of the associated vtree, as shown by [Xue et al., 2012]).

## References

- [Beame et al., 2013] Beame, P., Li, J., Roy, S., and Suci, D. (2013). Lower bounds for exact model counting and applications in probabilistic databases. In *UAI*, pages 157–162.
- [Beame et al., 2014] Beame, P., Li, J., Roy, S., and Suci, D. (2014). Counting of query expressions: Limitations of propositional methods. In *ICDT*, pages 177–188.
- [Breitbart et al., 1995] Breitbart, Y., Hunt III, H. B., and Rosenkrantz, D. J. (1995). On the size of binary decision diagrams representing boolean functions. *Theor. Comput. Sci.*, 145(1&2):45–69.
- [Dalvi and Suci, 2012] Dalvi, N. N. and Suci, D. (2012). The dichotomy of probabilistic inference for unions of conjunctive queries. *J. ACM*, 59(6):30.
- [Darwiche, 2001] Darwiche, A. (2001). Decomposable negation normal form. *J. ACM*, 48(4):608–647.

- [Darwiche, 2011] Darwiche, A. (2011). SDD: A new canonical representation of propositional knowledge bases. In *IJCAI 2011*, pages 819–826.
- [Gomes et al., 2009] Gomes, C. P., Sabharwal, A., and Selman, B. (2009). Model counting. In *Handbook of Satisfiability*, pages 633–654. IOS Press.
- [Huang and Darwiche, 2007] Huang, J. and Darwiche, A. (2007). The language of search. *JAIR*, 29:191–219.
- [Jha and Suciu, 2013] Jha, A. K. and Suciu, D. (2013). Knowledge compilation meets database theory: Compiling queries to decision diagrams. *Theory Comput. Syst.*, 52(3):403–440.
- [Kushilevitz and Nisan, 1997] Kushilevitz, E. and Nisan, N. (1997). *Communication Complexity*. Cambridge University Press, Cambridge, England ; New York.
- [Pipatsrisawat and Darwiche, 2008] Pipatsrisawat, K. and Darwiche, A. (2008). New compilation languages based on structured decomposability. In *AAAI*, pages 517–522.
- [Pipatsrisawat and Darwiche, 2010] Pipatsrisawat, K. and Darwiche, A. (2010). A lower bound on the size of Decomposable Negation Normal Form. In *AAAI*, pages 345–350.
- [Pipatsrisawat, 2010] Pipatsrisawat, T. (2010). *Reasoning with Propositional Knowledge: Frameworks for Boolean Satisfiability and Knowledge Compilation*. PhD thesis, UCLA.
- [SDD, 2014] SDD (2014). The SDD Package: Version 1.1.1. <http://reasoning.cs.ucla.edu/sdd/>.
- [Van den Broeck and Darwiche, 2015] Van den Broeck, G. and Darwiche, A. (2015). On the role of canonicity in knowledge compilation. In *AAAI*, pages 1641–1648.
- [Wegener, 2000] Wegener, I. (2000). *Branching programs and binary decision diagrams: theory and applications*. SIAM, Philadelphia, PA, USA.
- [Xue et al., 2012] Xue, Y., Choi, A., and Darwiche, A. (2012). Basing decisions on sentences in decision diagrams. In *AAAI*, pages 842–849.
- [Yannakakis, 1991] Yannakakis, M. (1991). Expressing combinatorial optimization problems by linear programs. *Journal of Computer and System Sciences*, 43(3):441–466.

---

# Hashing-Based Approximate Probabilistic Inference in Hybrid Domains

---

**Vaishak Belle**

Dept. of Computer Science  
KU Leuven  
Belgium  
vaishak@cs.kuleuven.be

**Guy Van den Broeck**

Dept. of Computer Science  
KU Leuven  
Belgium  
guy.vandenbroeck@cs.kuleuven.be

**Andrea Passerini**

DISI  
University of Trento  
Italy  
passerini@disi.unitn.it

## Abstract

In recent years, there has been considerable progress on fast randomized algorithms that approximate probabilistic inference with tight tolerance and confidence guarantees. The idea here is to formulate inference as a counting task over an annotated propositional theory, called weighted model counting (WMC), which can be partitioned into smaller tasks using universal hashing. An inherent limitation of this approach, however, is that it only admits the inference of discrete probability distributions. In this work, we consider the problem of approximating inference tasks for a probability distribution defined over discrete and continuous random variables. Building on a notion called weighted model integration, which is a strict generalization of WMC and is based on annotating Boolean and arithmetic constraints, we show how probabilistic inference in hybrid domains can be put within reach of hashing-based WMC solvers. Empirical evaluations demonstrate the applicability and promise of the proposal.

## 1 INTRODUCTION

Weighted model counting (WMC) on a propositional knowledge base is an effective and general approach to probabilistic inference in a variety of formalisms, including Bayesian and Markov Networks. It extends the model counting task, or #SAT, which is to count the number of assignments (that is, models) that satisfy a given logical sentence (Gomes *et al.*, 2009). In WMC, one accords a weight to every model, and computes the sum of the weights of all models. The WMC formulation has recently emerged as an assembly language for probabilistic reasoning, offering a basic formalism for encoding various inference problems. State-of-the-art reasoning algorithms for Bayesian networks (Chavira and Darwiche, 2008), their relational

extensions (Chavira *et al.*, 2006), factor graphs (Choi *et al.*, 2013), probabilistic programs (Fierens *et al.*, 2013), and probabilistic databases (Suciu *et al.*, 2011) reduce their inference problem to a WMC computation. The task has been generalized to first-order knowledge bases as well (Van den Broeck *et al.*, 2011; Gogate and Domingos, 2011). Exact WMC solvers are based on knowledge compilation (Darwiche, 2004; Muise *et al.*, 2012) or DPLL search with component caching (Sang *et al.*, 2005).

However, exact inference is #P-hard (Valiant, 1979), and so, there is a growing interest in approximate model counters. Beginning with Stockmeyer (1983), who showed that approximating model counting with a tolerance factor can be achieved in deterministic polynomial time using a  $\Sigma_2^P$ -oracle, a number of more recent results show how random polynomial-time realizations are possible using an NP-oracle (e.g., a SAT solver) (Jerrum *et al.*, 1986; Karp *et al.*, 1989; Bellare *et al.*, 2000; Gomes *et al.*, 2006; Ermon *et al.*, 2013b, 2014; Chakraborty *et al.*, 2013a,b). The central idea here is the use of random parity constraints, in the form of *universal hash functions* (Sipser, 1983), that partition the model counting solution space in an inexpensive manner. Most of the recent work in the area, moreover, come with strong tolerance-confidence guarantees (introduced later), and scale well by leveraging SAT technology.

The popularity of WMC can be explained as follows. Its formulation elegantly decouples the logical or symbolic representation from the statistical or numeric one, which is encapsulated in the weight function. When building solvers, this allows us to reason about logical equivalence and reuse SAT solving technology (such as constraint propagation and clause learning). WMC also makes it more natural to reason about deterministic, hard constraints in a probabilistic context. Nevertheless, WMC has a fundamental *limitation*: it is purely Boolean. This means that the advantages mentioned above only apply to *discrete probability distributions*.

To counter this, in a companion paper (Belle *et al.*, 2015), we proposed the notion of *weighted model integration* (WMI). It is based on *satisfiability modulo theories* (SMT),

which enable us to, for example, reason about the satisfiability of linear constraints over the rationals. The WMI task is defined on the models of an SMT theory  $\Delta$ , containing mixtures of Boolean and continuous variables. For every assignment to the Boolean and continuous variables, the WMI problem defines a weight. The total WMI is computed by integrating these weights over the domain of solutions of  $\Delta$ , which is a mixed discrete-continuous space. Consider, for example, the special case when  $\Delta$  has no Boolean variables, and the weight of every model is 1. Then, the WMI simplifies to computing the volume of the polytope encoded in  $\Delta$ . Overall, weighted SMT theories admit a natural encoding of hybrid Markov and Bayesian networks, analogous to the encodings of discrete graphical networks using weighted propositional theories.

In this work, we consider the problem of approximating inference tasks for a probability distribution defined over discrete and continuous random variables. Formulated as a WMI task, we address the question as to whether fast hashing-based approximate WMC solvers can be leveraged for hybrid domains. What we show is that an NP-oracle can indeed effectively partition the model counting solution space of the more intricate mixed discrete-continuous case using universal hashing. (Of course, volume computation is still necessary, but often over very small spaces.) In this sense, hybrid domains can now be put within reach of approximate WMC solvers. In particular, the hashing approach that we consider here builds on the recent work of Chakraborty *et al.* (2014) on approximate WMC, and inherits their tolerance-confidence guarantees. In our empirical evaluations, the approximate technique is shown to be significantly faster than an exact WMI solver. We then demonstrate the practical efficacy of the system on a complex real-world dataset where we compute conditional queries over intricate arithmetic constraints that would be difficult (or impossible) to realize in existing formalisms.

Let us finally mention that current inference algorithms for hybrid graphical models often make strong assumptions on the form of the potentials, such as Gaussian distributions (Lauritzen and Jensen, 2001), or approximate using variational methods (Murphy, 1999; Lunn *et al.*, 2000), for which quality guarantees are difficult to obtain. There is also a recent focus on *piecewise-polynomial* potentials (Shenoy and West, 2011; Sanner and Abbasnejad, 2012; Wang *et al.*, 2014), which are based on generalizations of techniques such as the join-tree algorithm. Such piecewise-polynomials can also be represented in the WMI context, but in a general framework allowing arbitrary Boolean connectives and deterministic hard constraints.

## 2 PRELIMINARIES

We begin with probabilistic models, and then turn to the necessary logical background, WMC and WMI.

### 2.1 PROBABILISTIC MODELS

Let  $\mathcal{B}$  and  $\mathcal{X}$  denote sets of Boolean and real-valued random variables, that is,  $b \in \mathcal{B}$  is assumed to take values from  $\{0, 1\}$  and  $x \in \mathcal{X}$  takes values from  $\mathbb{R}$ . We let  $(\mathbf{b}, \mathbf{x}) = (b_1, \dots, b_m, x_1, \dots, x_n)$  be an element of the probability space  $\{0, 1\}^m \times \mathbb{R}^n$ , which denotes a particular assignment to the random variables from their respective domains. We let the joint probability density function be denoted by  $\text{Pr}$ . So  $\text{Pr}(\mathbf{b}, \mathbf{x})$  determines the probability of the assignment vector. When these random variables are defined by a set of dependencies, as can be represented using an *undirected graphical model* (that is, Markov network), the density function is compactly *factorized*. See Koller and Friedman (2009) for details.

### 2.2 LOGICAL BACKGROUND

*Propositional satisfiability* (SAT) is the problem of deciding whether a logical formula over Boolean variables and logical connectives can be satisfied by some truth value assignment of the Boolean variables. Given a formula  $\phi$  and assignment (or model or world)  $M$ , we write  $M \models \phi$  to denote *satisfaction*. We write  $l \in M$  to denote the literals (that is, propositions or their negations) that are satisfied at  $M$ . We often write  $\mathcal{M}(\phi)$  to mean the set of models of  $\phi$ .

A generalization to this decision problem is that of *Satisfiability Modulo Theories* (SMT). In SMT, we are interested in deciding the satisfiability of a (typically quantifier-free) first-order formula with respect to some decidable background theory  $\mathcal{T}$ , such as linear arithmetic over the rationals ( $\mathcal{LRA}$ ). Standard first-order models can be used to formulate SMT; see Barrett *et al.* (2009) for details. Moreover various background theories, like  $\mathcal{LRA}$  and linear arithmetic over the integers ( $\mathcal{LIA}$ ), can be combined. In this paper we are interested in a combination of  $\mathcal{LRA}$  and propositional logic, for which satisfaction is defined in an obvious way.

Our formulation will also use the concepts of *formula abstraction* and *refinement* (Barrett *et al.*, 2009). Here, first, a bijection is established between ground first-order atoms and a propositional vocabulary; abstraction proceeds by replacing the atoms by propositions, and refinement replaces the propositions with the atoms. In the sequel, we refer to the propositional abstraction of an SMT formula  $\phi$  as  $\phi^-$  and the refinement of  $\phi$  as  $\phi^+$ . For example, if  $\Delta = (x \leq 4) \wedge (x \leq 5)$ , then  $\Delta^- = p \wedge q$  where (say)  $p$  denotes  $x \leq 4$  and  $q$  denotes  $x \leq 5$ ; also,  $q^+ = x \leq 5$ .

### 2.3 WEIGHTED MODEL COUNTING

Weighted model counting (Chavira and Darwiche, 2008) is an extension of model counting (Gomes *et al.*, 2009). In model counting, also known as #SAT, one counts the number of satisfying assignments of a propositional sentence.



In WMC, each assignment has an associated weight and the task is to compute the sum of the weights of all satisfying assignments. WMC has applications in probabilistic inference in discrete graphical models.

**Definition 1:** Given a formula  $\Delta$  in propositional logic over literals  $\mathcal{L}$ , and a *weight function*  $w : \mathcal{L} \rightarrow \mathbb{R}$ , the *weighted model count* (WMC) is defined as:

$$\text{WMC}(\Delta, w) = \sum_{M \models \Delta} w(M)$$

where,  $w(M)$  is shorthand for  $\prod_{l \in M} w(l)$ .

Intuitively, the weight of a formula is given in terms of the total weight of its models; the weight of a model is defined in terms of the literals true in that model.

We are often interested in computing the probability of a query  $q$  given evidence  $e$  in a Boolean Markov network  $N$ , for which we use:

$$\Pr_N(q \mid e) = \frac{\text{WMC}(q \wedge e \wedge \Delta, w)}{\text{WMC}(e \wedge \Delta, w)}$$

where  $\Delta$  encodes  $N$  and  $w$  encodes the potentials; see, for example, Chavira and Darwiche (2008).

## 2.4 WEIGHTED MODEL INTEGRATION

As noted before, an inherent limitation of WMC is that it only admits the inference of discrete probability distributions. To remedy this, in a companion paper (Belle *et al.*, 2015), we introduced the notion of *weighted model integration* as a strict generalization of WMC. The main idea here is to take a logical theory with rational and Boolean variables, that is, from a combination of  $\mathcal{LR}\mathcal{A}$  and propositional logic, and annotate it with weights. As before, propositional assignments are denoted using  $M$ .

**Definition 2:** Suppose  $\Delta$  is an SMT theory over Boolean and rational variables  $\mathcal{B}$  and  $\mathcal{X}$ , and literals  $\mathcal{L}$ . Suppose  $w : \mathcal{L} \rightarrow \text{EXPR}(\mathcal{X})$ , where  $\text{EXPR}(\mathcal{X})$  are expressions over  $\mathcal{X}$ . Then the *weighted model integral* (WMI) is defined as:

$$\text{WMI}(\Delta, w) = \sum_{M \models \Delta^-} \text{VOL}(M, w)$$

$$\text{where, } \text{VOL}(M, w) = \int_{\{l^+ : l \in M\}} w(M) d\mathcal{X}.$$

The main feature of the definition is how it casts the weighted model counting problem over SMT in standard propositional logic. The intuition is as follows. The WMI of an SMT theory  $\Delta$  is defined in terms of the models of its propositional abstraction  $\Delta^-$ . For each such model, we compute its volume, that is, we integrate the weight values of the literals that are true at the model. The interval of

the integral is obtained from the refinement of each literal.<sup>1</sup> The mathematical expression for conditional probabilities is as before.

The general idea with  $\text{EXPR}(\mathcal{X})$  is that the weight function maps an expression  $e$  to its *density function*, which is usually another expression mentioning the variables in  $e$ . We note that the input language for a WMI task is easily seen to capture constraints involving discrete and continuous random variables over arbitrary Boolean connectives.

To see WMI in action, consider a simple example:

**Example 3:** Suppose  $\Delta$  is the following formula:

$$p \vee (0 \leq x \leq 10)$$

For weights, let  $w(p) = .1$ ,  $w(\neg p) = 2x$ ,  $w(q) = 1$  and  $w(\neg q) = 0$ , where  $q$  is the propositional abstraction of  $(0 \leq x \leq 10)$ . Roughly, this can be seen to say that  $x$  is uniformly distributed when  $p$  holds and otherwise it is characterized by a triangular distribution in the interval  $[0, 10]$ . There are three models of  $\Delta^-$ , for which we calculate  $\text{VOL}(\cdot, w)$ :

1.  $\text{VOL}(\{p, \neg q\}, w) = 0$  because  $w(\neg q) = 0$ ;
2.  $\text{VOL}(\{\neg p, q\}, w) = \int_{0 \leq x \leq 10} 2x dx = [x^2]_0^{10} = 100$ .
3.  $\text{VOL}(\{p, q\}, w) = \int_{0 \leq x \leq 10} .1 dx = [.1 \cdot x]_0^{10} = 1$ .

Thus,  $\text{WMI}(\Delta, w) = 100 + 1 = 101$ .

Suppose that we are interested in the probability of the query  $x \leq 3$  given that  $\neg p$  is observed. Suppose  $r$  is the abstraction of  $x \leq 3$ . First,  $\text{WMI}(\Delta \wedge \neg p, w)$  corresponds to the weight of a single interpretation, that of item 2, yielding a value of 100. Next,  $\text{WMI}(\Delta \wedge \neg p \wedge x \leq 3, w)$  also corresponds to the weight of a single interpretation  $\{\neg p, q, r\}$ , an extension to that in item 2. In this case:

$$\text{VOL}(\{\neg p, q, r\}, w) = \int_{(0 \leq x \leq 10) \wedge (x \leq 3)} 2x dx = [x^2]_0^3 = 9.$$

Therefore, the conditional probability is  $9/100 = .09$ .  $\square$

<sup>1</sup>Although the interval is defined in terms of SMT literals, this is meant to denote standard integrals in an obvious fashion:

$$\int_{x \leq 6} \phi dx \doteq \int_{-\infty}^6 \phi dx; \int_{x \geq 6} \phi dx \doteq \int_6^{\infty} \phi dx; \int_{5 \leq x \leq 6} \phi dx \doteq \int_5^6 \phi dx$$

Likewise, over connectives:

$$\int_{x \leq 6 \wedge y \geq 5} \phi dx dy \doteq \int_{x \leq 6} \int_{y \geq 5} \phi dx dy.$$

When propositions appear as intervals, they are simply ignored. See Belle *et al.* (2015) for the general definition.

The correctness of WMI and that it is a strict generalization of WMC are argued elsewhere (Belle *et al.*, 2015).<sup>2</sup>

Let us conclude this section by remarking that although the definition of WMI is very general, for most practical purposes, we restrict densities to piecewise polynomials, where  $w$  maps  $\mathcal{L}$  to polynomials over  $X$ . Such piecewise polynomials can approximate a wide variety of continuous distributions, including Gaussians (Shenoy and West, 2011; Sanner and Abbasnejad, 2012). Moreover, Baldoni *et al.* (2011) show that for a fixed number of variables, the integration is efficient, even for polynomials of increasing degree. Thus, smooth function approximations are possible in practice, and come at a reasonable cost.

### 3 APPROXIMATING WMI

In this section, we identify how to approximate  $\text{WMI}(\Delta, w)$  for an arbitrary  $\Delta$  and non-degenerate (see below)  $w$  with strong theoretical guarantees by appealing to a SAT-oracle.

#### 3.1 PROBLEM STATEMENT FOR WMC

To better understand the problem statement, let us begin with the case of WMC:

**Definition 4:** Given a propositional formula  $\Delta$  and a weight function  $w$ , an *exact algorithm* for WMC returns  $\text{WMC}(\Delta, w)$ . An *approximate algorithm* for WMC given *tolerance*  $\epsilon \in (0, 1]$  and *confidence*  $1 - \delta \in (0, 1]$ , simply called an  $(\epsilon, \delta)$ -algorithm, returns a value  $v$  such that

$$\Pr \left[ \frac{\text{WMC}(\Delta, w)}{1 + \epsilon} \leq v \leq (1 + \epsilon)\text{WMC}(\Delta, w) \right] \geq 1 - \delta$$

Intuitively, when the weight of every model is 1, an exact algorithm returns the size of the set  $\mathcal{M}(\Delta) = \{M \mid M \models \Delta\}$  while an approximate one samples from that solution space. Exact algorithms are #P-hard (Valiant, 1979) but for the approximate case random polynomial time realizations are known (Jerrum *et al.*, 1986; Karp *et al.*, 1989).<sup>3</sup>

<sup>2</sup>In an independent and recent effort, Chistikov *et al.* (2015) also introduce the notion of approximate model counting for SMT theories. The most significant difference between the proposals is that they focus only on unweighted model counting. Moreover, they define model counting as a measure on first-order models. Our approach is a simpler one based on propositional abstractions, which (as we will see) allows us to cast statements for WMI as WMC in a direct way.

<sup>3</sup>The class of  $(\epsilon, \delta)$ -algorithms that we are after follows the terminology of Karp *et al.* (1989). These can be contrasted to *bounding* counters only parameterized by confidence probabilities, such as (Kroc *et al.*, 2011).

#### 3.2 PROBLEM STATEMENT FOR WMI

To see how the above notions apply to our task, consider an SMT theory  $\Delta$  and weight function  $w$ . We observe that

$$\text{WMI}(\Delta, w) = \text{WMC}(\Delta^-, u)$$

where, for any model  $M$  of  $\Delta^-$ ,  $u$  is a weight function such that  $u(M) = \text{VOL}(M, w)$ . More precisely,  $u$  is to be seen as a weight function that does not factorize over literals and directly maps interpretations to  $\mathbb{R}$ . (This is without any loss of generality.) Thus, our problem statement becomes:

**Definition 5:** An  $(\epsilon, \delta)$ -algorithm for a WMI problem over  $\Delta$  and  $w$  is an  $(\epsilon, \delta)$ -algorithm for WMC over  $\Delta^-$  and weight function  $u$ , where for any model  $M$  of  $\Delta^-$ ,  $u(M) = \text{VOL}(M, w)$ .

The idea is that by treating the volumes of models as weights over propositional interpretations, we can view WMI simply in terms of WMC. Theoretical results can then be imported for our purposes.

Given an  $(\epsilon, \delta)$ -algorithm for WMC, there are two caveats, however. First, weights of interpretations need to be actually computed using integration during inference, but (usually) over a small number of literals and their polynomial potentials true in a model. Second, such an algorithm samples feasible satisfying assignments for  $\Delta^-$ , but these need not be  $\mathcal{T}$ -consistent. For example, if  $p$  denotes  $x \leq 3$  and  $q$  denotes  $x \leq 5$ , then the interpretation  $\{p, \neg q\}$  is not a model in  $\mathcal{LRA}$  on refinement. In the formal machinery, the weight of this model is easily seen to be 0 (that is, the interval of the integral will be an empty set), and so these models can freely appear in the problem statement. In practice, these theory inconsistency models are rejected in the WMC calculation, and once found, the algorithm can be made to refine its search of feasible solutions by incorporating these models as blocked clauses.

#### 3.3 APPROACH

In sum, what we are after is an  $(\epsilon, \delta)$ -algorithm for  $\text{WMC}(\Delta, w)$  for a propositional theory  $\Delta$  and weight function  $w$ . Consider classical model counting, that is, where the weight of every model is 1, which is #P-hard. Bellare *et al.* (2000) were the first to show that satisfying assignments can be generated uniformly in random polynomial-time using only an NP-oracle (e.g., a SAT solver), improving and complementing earlier results (Jerrum *et al.*, 1986; Karp *et al.*, 1989; Stockmeyer, 1983). Adapting these techniques further, Chakraborty *et al.* (2013a,b) introduce a scalable approximate model counter. See Gomes *et al.* (2006) and Ermon *et al.* (2013b) for closely related proposals.

### 3.3.1 Counting by Hashing

The central idea in Bellare *et al.* (2000) and Chakraborty *et al.* (2013b) is the use of *universal hash functions* (Sipser, 1983):

**Definition 6 :** A family of functions  $\mathcal{H} = \{h \mid \{0, 1\}^n \rightarrow \{0, 1\}^m\}$  is called *uniform*, written  $h \stackrel{R}{\leftarrow} \mathcal{H}$ , if it holds that for any  $y \in \{0, 1\}^m$ , the random variable  $h(y)$  is uniformly distributed in  $\{0, 1\}^n$ .

**Definition 7 :** A family of functions  $\mathcal{H} = \{h \mid \{0, 1\}^n \rightarrow \{0, 1\}^m\}$  is called *t-wise independent* if it holds that for any  $x_1, \dots, x_t \in \{0, 1\}^m$ , any  $y_1, \dots, y_t \in \{0, 1\}^n$ , and any  $h \stackrel{R}{\leftarrow} \mathcal{H}$ , we have:

$$\Pr[h(y_1) = x_1 \wedge \dots \wedge h(y_t) = x_t] = 2^{-m \cdot t}$$

For the sake of clarity, we denote this family as  $\mathcal{H}(n, m, t)$ . Now, suppose  $x \in \{0, 1\}^m$  and  $h \stackrel{R}{\leftarrow} \mathcal{H}(n, m, t)$ . Let  $h^{-1}(x) = \{y \in \{0, 1\}^n \mid h(y) = x\}$ . Then, the idea is that for any propositional language over  $n$  variables,  $\mathcal{M}(\Delta) \subseteq \{0, 1\}^n$  and  $x \in \{0, 1\}^m$ , the set  $\mathcal{M}(\Delta) \cap h^{-1}(x)$  partitions  $\mathcal{M}(\Delta)$  into a set of well-balanced *cells*, each one induced by a particular choice of  $h$ . Thus, by iterating over different samples of  $h \stackrel{R}{\leftarrow} \mathcal{H}(n, m, t)$ , we can perform a small number of computations on the cells and leverage that as an estimate for the solution space as a whole.

The work of Chakraborty *et al.* (2013b) uses an efficient family of hash functions, denoted  $\mathcal{H}_{\text{xor}}(n, m, 3)$  below. Given  $h \stackrel{R}{\leftarrow} \mathcal{H}(n, m, 3)$  and  $y \in \{0, 1\}^n$ , let  $h(y)[k]$  denote the  $k$ th component of the vector obtained by applying  $h$  to  $y$ , and  $y[k]$  denote the  $k$ th component of the string  $y$ . The family of hash functions of interest is defined as

$$\mathcal{H}_{\text{xor}}(n, m, 3) = \{h \mid h(y)[i] = a_{i,0} \oplus \left(\bigoplus_{l=1}^n a_{i,l} \cdot y[l]\right), \\ a_{i,j} \in \{0, 1\}, 1 \leq i \leq m, 0 \leq j \leq n\}$$

where  $\oplus$  denotes the XOR operation. By choosing values of  $a_{i,j}$  randomly and independently, we can effectively choose a random hash function from the family. Gomes *et al.* (2006) show that this family of hash functions is 3-wise independent.

### 3.3.2 WMC by Hashing

As argued by Ermon *et al.* (2013a), the one major limitation when applying approximate model counters for probabilistic inference is that weights play an important role in deeming which samples are interesting. Therefore, uniformly sampling from  $\mathcal{M}(\phi)$  is not appealing, and would lead to poor estimates of conditional probabilities. The approach taken in Ermon *et al.* (2013a) is to reformulate the inference task by an embedding for which uniform sampling suffices.

While this is an attractive option, it requires a factored representation of the probability distribution and appeals to solutions of optimization problems from a MPE query (that is, finding the most likely state). In our setting, these requirements are problematic. For one thing, note that even if the original input problem uses a factored representation, we only possess a WMC problem with a weight function for interpretations that (possibly) lacks any structure. For another, MPE queries will involve computing integrals (recall that weights are computed during inference) for a large number of states, a task we would like to avoid unless necessary.

Nonetheless, extending earlier results (Bellare *et al.*, 2000; Chakraborty *et al.*, 2013b), Chakraborty *et al.* (2014) (CFMSV henceforth) show how approximate model counters can be applied to weighted model counting problems by means of a parameter called *tilt*.

**Definition 8:** Suppose  $\Delta$  is a propositional theory and  $w$  is a weight function mapping  $\mathcal{M}(\Delta)$  to strictly positive numbers. Let  $w_{\max} = \max_M w(M)$  and let  $w_{\min} = \min_M w(M)$ . We define the *tilt*  $\theta$  to be the ratio  $w_{\max}/w_{\min}$ .

For our purposes, we adapt the notion as follows:

**Definition 9:** Suppose  $\Delta$  is a SMT theory over literals  $\mathcal{L}$  and continuous variables  $\mathcal{X}$  and  $w$  is a weight function mapping  $\mathcal{L}$  to  $\text{EXPR}(\mathcal{X})$ . Let  $w_{\max} = \max_M \text{VOL}(M, w)$  and let  $w_{\min} = \min_M \text{VOL}(M, w)$ . We define the *tilt*  $\theta$  to be the ratio  $w_{\max}/w_{\min}$ .

The idea is that in approximate model counting, the number of hash functions to sample (and thus, the number of cells to construct of  $\mathcal{M}(\Delta)$ ) is guided by the confidence parameter  $\delta$ . In the approach of Chakraborty *et al.* (2014), the tilt of a problem is used to additionally ensure that an appropriate number of cells are constructed in the weighted case. While the approach requires the modeler to provide an upper bound on the tilt, the parameter is agnostic about the form of the weight function, which is desirable in our setting. Nonetheless, when the tilt is large (i.e., when the weight of an interpretation is small relative to others), it would mean that a large number of cells are constructed, which may be inefficient, and alleviating this is an interesting avenue for the future. In practice, the problems we encountered had small tilts. (In our experimental evaluations, an upper bound of 5-10 was provided for the tilt.)

## 3.4 ALGORITHM

Putting it all together, we present the pseudocode for WMI computation. It can be seen as a simple reworking of CFMSV to solve WMI.<sup>4</sup> For the sake of completeness, we

<sup>4</sup>In that sentiment, we believe an important feature of our formulation is that other WMC approaches can be adapted for WMI along the same lines.

---

**Algorithm 1** WEIGHTMC( $\phi, u, \epsilon, \delta, \theta$ )

---

```
1:  $w_{\max} \leftarrow 1$ 
2:  $\text{pivot} \leftarrow 2 \times \lceil e^{3/2} \left(1 + \frac{1}{\epsilon}\right)^2 \rceil$ 
3:  $\text{iter} \leftarrow \lceil 35 \log_2(3/\delta) \rceil$ 
4: for  $i : 1, \dots, \text{iter}$  do
5:    $(\mathcal{M}, c, w_{\max}) \leftarrow \text{WEIGHTMCCORE}(\phi, u, \text{pivot}, \theta, w_{\max})$ 
6:   store  $(c, w_{\max})$  if  $\mathcal{M} \neq \emptyset$ 
7: end for
8: return the median of  $c \times w_{\max}$  for stored tuples
```

---

---

**Algorithm 2** WEIGHTMCCORE( $\phi, u, \text{pivot}, \theta, w_{\max}$ )

---

```
1:  $i \leftarrow 0$ ;  $\text{vol} \leftarrow 0$ ;  $n \leftarrow$  number of variables in  $\phi$ 
2: repeat
3:    $i \leftarrow i + 1$ 
4:   Choose  $h \xleftarrow{R} \mathcal{H}_{\text{xor}}(n, i, 3)$ 
5:   Choose  $x \xleftarrow{R} \{0, 1\}^i$ 
6:    $(\mathcal{M}, \text{vol}, w_{\max}) \leftarrow \text{BOUNDEDWEIGHTSAT}(\phi, (h(b_1, \dots, b_n) = x), u, \text{pivot}, \theta, w_{\max})$ 
7: until  $(0 < \text{vol}/w_{\max} \leq \text{pivot}$  or  $i = n)$ 
8: if  $(\text{vol}/w_{\max} > \text{pivot}$  or  $\mathcal{M} = \emptyset)$  then return  $(\emptyset, \text{vol}, w_{\max})$ 
9: else return  $(\mathcal{M}, \text{vol} \cdot 2^{i-1}/w_{\max}, w_{\max})$ 
10: end if
```

---

present the essential components of the CFMSV algorithm, called WEIGHTMC. Interested readers are referred to that work for full details. First, given an SMT theory  $\Delta$ , weight function  $w$ , tolerance  $\epsilon$ , confidence  $\delta$  and an upper bound on the tilt  $\theta$ , we compute:<sup>5</sup>

$$\text{WMI}(\Delta, w, \epsilon, \delta, \theta) = \text{WEIGHTMC}(\Delta^-, u, \epsilon, \delta, \theta)$$

where  $u$  is the weight function mapping interpretations to numbers and is calculated using:

$$u(M) = \text{VOL}(M, w).$$

The WEIGHTMC procedure is given in Algorithm 1. Basically, the given parameters  $\delta$  and  $\epsilon$  are used to determine the number of times WEIGHTMCCORE is invoked and the number of cells to induce on  $\mathcal{M}(\Delta^-)$ , respectively. What the procedure returns is the median of the volume estimates, obtained from WEIGHTMCCORE over these iterations. For any given iteration, if no model is found, then the estimates from WEIGHTMCCORE are ignored.

The procedure WEIGHTMCCORE applied to a propositional formula  $\phi$ , detailed in Algorithm 2, partitions models of  $\phi$  into cells. This is achieved by choosing 3-wise independent hash functions, and adding random parity constraints. The resulting logical formula is conjoined with

---

<sup>5</sup>CFMSV's WEIGHTMC procedure also includes a parameter called the *independent support* of a propositional theory  $\phi$  over variables  $\mathcal{B}$ . The support of  $\phi$  is  $\mathcal{B}$ , and the independent support  $\mathcal{I} \subseteq \mathcal{B}$  uniquely determine the truth values of variables from  $\mathcal{B} - \mathcal{I}$ . By choosing hash functions only on the independent support, rather than the full set of variables in  $\mathcal{B}$ , significant performance improvements can be gained. But since this is inessential to understanding the conceptual ideas of the algorithm, we omit further discussion on this matter.

---

**Algorithm 3** BOUNDEDWEIGHTSAT( $\phi, \chi, u, \text{pivot}, \theta, w_{\max}$ )

---

```
1:  $w_{\min} \leftarrow w_{\max}/\theta$ ;  $\text{vol} \leftarrow 0$ ;  $\mathcal{M} = \emptyset$ ;  $\gamma = \phi \wedge \chi$ 
2: repeat
3:    $M \leftarrow \text{SOLVESAT}(\gamma)$ 
4:   if  $M = \text{UNSAT}$  then
5:     break
6:   end if
7:    $\text{CONS} \leftarrow \text{CONSISTENT}(\mathcal{T}, \{l^+ \mid l \in M\})$ 
8:   if  $\text{CONS} = \text{INCONSISTENT}$  then
9:      $\phi \leftarrow \text{ADDBLOCKCLAUSE}(\phi, M)$ 
10:  else
11:     $\mathcal{M} \leftarrow \mathcal{M} \cup M$ 
12:     $\gamma \leftarrow \text{ADDBLOCKCLAUSE}(\gamma, M)$ 
13:     $\text{CACHE}[M] \leftarrow u(M)$  if  $\text{CACHE}[M] = \{\}$ 
14:     $\text{vol} \leftarrow \text{vol} + \text{CACHE}[M]$ 
15:     $w_{\min} \leftarrow \min(w_{\min}, \text{CACHE}[M])$ 
16:  end if
17: until  $\text{vol}/(w_{\min} \cdot \theta) > \text{pivot}$ 
18: return  $(\mathcal{M}, \text{vol}, w_{\min} \cdot \theta)$ 
```

---

$\phi$ , for which BOUNDEDWEIGHTSAT is invoked. The number of iterations of BOUNDEDWEIGHTSAT is bound by the number of propositional variables in  $\phi$ , or when the current tilt exceeds an  $\epsilon$ -based parameter. Among other things, BOUNDEDWEIGHTSAT returns the models of  $\phi$  conjoined with a random parity constraint, and unless this is empty, volume estimates in WEIGHTMCCORE are returned to WEIGHTMC. Both WEIGHTMCCORE and WEIGHTMC are minor adaptations of the procedures from CFMSV in the following sense: in the CFMSV modules, the weights are directly used; for example, line 7 in Algorithm 2 would explicitly refer to  $u(\mathcal{M}) = \sum_{M \in \mathcal{M}} u(M)$ . In our setting, computing  $u(\mathcal{M})$  involves integration which we would not want to repeat for every iteration. Thus, weights of (sets of) models are themselves returned when required. It is easy to see that the analysis of these procedures is unaffected by this adaptation.

Finally, we turn to BOUNDEDWEIGHTSAT in Algorithm 3, where a more significant adaptation of the CFMSV scheme occurs. First, one would observe that, different from CFMSV, the parity constraint  $\chi$  is provided separate from the input formula (for reasons justified below). Nonetheless, the procedure essentially performs a bounded version of model counting on  $\gamma = \phi \wedge \xi$ , as would CFMSV, where a  $(\theta, \epsilon)$ -derived parameter determines this bound. As soon as no model is found, the procedure exits with the current estimates. If a model  $M$  is found, however, unlike CFMSV, we need to additionally ensure the refinement of this assignment is consistent w.r.t. the background theory  $\mathcal{T}$ . If it is not  $\mathcal{T}$ -consistent, then we can prevent this model from being considered for all iterations by adding it as a block clause to our input propositional formula  $\phi$ . (Recall also that such a model would have zero volume, leading to an infinite tilt if it were to be considered.) On the other hand, if it is theory consistent, we compute its volume and then add it as a block clause to  $\gamma$ . (This achieves the model counting of  $\gamma$ .) In particular, we calculate  $u(M) = \text{VOL}(M, w)$  where

$w$  is the actual weight function from the WMI problem, and cache this result. So, integration is performed only once for the model  $M$ . The procedure then returns the total volume of the set of models  $\mathcal{M}$  identified.

What is perhaps interesting to realize of this adaptation of CFMSV is that it also does not affect the analysis of the procedures. Note that, if the problem instance is a propositional theory, then  $\Delta^- = \Delta$ , and line 7 of Algorithm 3 is trivially true because  $\mathcal{T}$  is propositional logic. Consequently, the test in line 8 is trivially false. More formally:

**Proposition 10:** *Suppose  $\phi$  is a propositional formula,  $u$  is a weight function,  $\epsilon, \delta \in (0, 1]$ , and  $\theta$  is an upper bound on the tilt. Then  $\text{WEIGHTMC}(\phi, u, \epsilon, \delta, \theta)$  from CFMSV’s original formulation returns  $c$  iff the current adaptation does.*

This allows us, very easily, to leverage the strong guarantees of CFMSV (our rewording):

**Theorem 11:** [CFMSV] *Suppose  $\phi, u, \epsilon, \delta, \theta$  are as above. Then  $\text{WEIGHTMC}(\phi, u, \epsilon, \delta, \theta)$  is an  $(\epsilon, \delta)$ -algorithm for  $\text{WMC}(\phi, u)$ . Given a SAT-oracle, it runs in time polynomial in  $\log_2(1/\delta), \theta, |\phi|$  and  $1/\epsilon$  relative to the oracle.*

From which we obtain:

**Corollary 12:** *Suppose  $\Delta$  is an SMT theory,  $w$  is a weight function, and  $\epsilon, \delta, \theta$  are as above. Suppose  $u$  is the derived weight function for  $\Delta^-$ . Then,  $\text{WEIGHTMC}(\Delta^-, u, \epsilon, \delta, \theta)$  is an  $(\epsilon, \delta)$ -algorithm for  $\text{WMI}(\Delta, w)$ . Suppose we are given an oracle to the weight function  $u$  and a SAT-oracle. Then,  $\text{WEIGHTMC}(\Delta^-, u, \epsilon, \delta, \theta)$  runs in time polynomial in  $\log_2(1/\delta), \theta, |\Delta^-|$  and  $1/\epsilon$  relative to the oracles.*

Thus, we can inherit existing results for this WMC solver (and perhaps others). The oracle to  $u$  computes the volumes of  $\mathcal{T}$ -consistent models. Each instance of  $\text{VOL}(M, w)$  involves the following (Belle *et al.*, 2015):

**Proposition 13:** *Suppose  $\Delta$  is an SMT theory over continuous variables  $\mathcal{X}$ ,  $M$  a model of  $\Delta^-$ , and  $w$  is as above. Let  $k$  be the maximum degree of the polynomials in  $\{w(l) \mid l \in M\}$ . Then  $\text{VOL}(M, w)$  integrates polynomials of degree  $k \cdot |M|$ .*

Basically, for any model  $M$  of  $\Delta^-$ ,  $\text{VOL}(M, w)$  is formulated as the integration of the polynomial potentials of the literals true at  $M$ , which would be a product of  $|M|$  polynomials, each of degree  $k$ . As mentioned earlier, Baldoni *et al.* (2011) show that when the number of variables is fixed (as determined by  $|\mathcal{X}|$ ), integration is efficient. In practice, moreover, when encoding factored representations and piecewise polynomials, it is often the case that negated atoms are assigned constant weights and so we encounter polynomials of degree  $k \cdot n$  where  $n \ll |M|$ . In essence, what we usually encounter are a small number of polynomial potentials corresponding to atoms that are true in the model.

## 4 EMPIRICAL EVALUATIONS

In this section, we discuss results on an implementation of the approximate inference system. The inference system builds on an implementation of  $\text{WEIGHTMC}$  from CFMSV, which uses  $\text{CRYPTOMINISAT v2}$  to handle XOR clauses efficiently.<sup>6</sup> The  $\text{BOUNDEDWEIGHTSAT}$  module appeals to the Z3 SMT solver v4.3.2 for testing theory consistency,<sup>7</sup> and the  $\text{LATE}$  software v1.6 for computing integrals.<sup>8</sup> All experiments were run using a system with 1.7 GHz Intel Core i7 and 8GB RAM. Moreover, to maintain consistency with CFMSV and their parameterization, experiments were run for  $\epsilon = .8, \delta = .2$  and  $\theta$  was assumed to be 5.

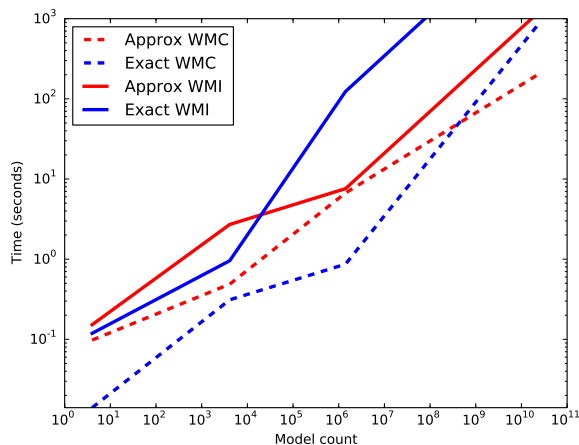


Figure 1: scaling behavior

### 4.1 SCALING BEHAVIOR

In large domains over arbitrary polynomial potentials, approximate inference can be assumed to offer significant savings. To test this, we used a benchmark from our prior work (Belle *et al.*, 2015). In that work, we randomly generated SMT theories and weight functions, involving intricate dependencies and hard constraints. These included weighted sentences of the form:

$$\begin{aligned} x^3 & f_1 \Leftrightarrow [x + 3y \leq 3] \\ .001y^2 - .11y + 2.5 & f_2 \Leftrightarrow [p \vee (x \geq 0 \wedge y \geq 0)] \\ .1 & f_3 \Leftrightarrow (\neg p \vee \neg q) \end{aligned}$$

(that is, the LHS is the weight function for the expression in the RHS) with additional hard constraints of the sort:

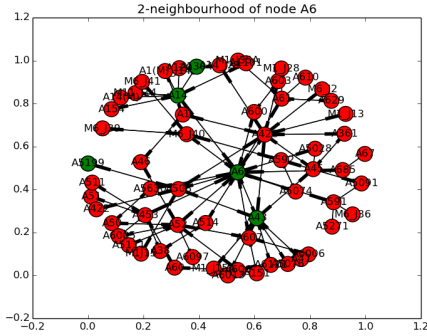
$$(f_1 \wedge f_2) \Rightarrow \neg f_3.$$

In our prior work, an exact WMI solver was implemented using a block-clause strategy: in each iteration, if a theory-consistent model is found, a clause over the negations of

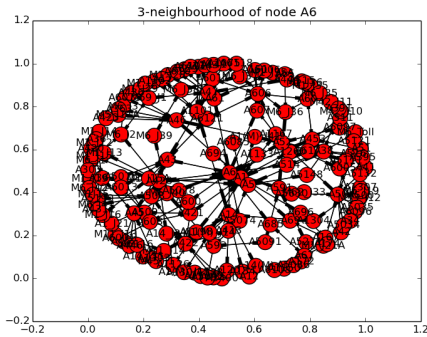
<sup>6</sup><http://www.msoos.org/cryptominisat2/>

<sup>7</sup><http://z3.codeplex.com>

<sup>8</sup><https://www.math.ucdavis.edu/~latte>



(a) at most 2 junctions



(b) at most 3 junctions

Figure 2: Strategic Road Network portions surrounding motorway A6.

the literals in the assignment is added as an additional constraint, and in this way, all models are enumerated. Using the above benchmark weighted SMT theories, we plot the approximate WMI system against the exact WMI implementation. To further contrast this to the simpler setting of classical propositional logic, we also ran experiments for an exact WMC and the approximate WMC of CFMSV for the input problem  $\Delta^-$  (that is, the propositional abstraction). Figure 1 illustrates that for small problems, exact computations are usually faster, both in the propositional and SMT setting, the former observation already reported in CFMSV. In large domains, however, exact WMC already fares poorly against the approximate version, and in the WMI setting, exact computations become infeasible, and cannot compete with the approximate module for increasing problem sizes.

## 4.2 REAL-WORLD DATASET

To demonstrate the expressivity and usefulness of the approach in a complex real-world scenario, we consider the following novel application involving conditional queries over arithmetic constraints. It uses a data series released by the UK government that provides average journey time,

speed and traffic flow information on all motorways, known as the Strategic Road Network, in England.<sup>9</sup> Motorways are split into junctions, and each information record refers to a specific junction, day and time period. In the following we consider the 2012 dataset, with over 7 million entries, and focus on the surroundings of the A6 motorway. Figures 2a and 2b show the portion of the network with at most two and three junctions respectively from A6. We extract statistics on journey time across each junction. For the sake of simplicity, we model a junction’s journey time as a uniform distribution between the observed minimum and maximum travel time.

Consider a planning problem for a supply system for motorway service stations. The operations center (located, say, somewhere along A6) receives supply requests from a number of stations, and needs to predict whether the delivery vehicle will be able to reach all stations and return within a certain amount of time. Travel time between every pair of stations, and between stations and the operations center, is computed in terms of shortest paths across the network. We compute shortest paths for both minimum and maximum travel times, so as to get a distribution for the shortest path duration w.r.t. every pair of relevant points (stations and operations center), which, as noted, is uniform between these two extremes. Given a certain route between stations, the probability of completing it within the desired time can be computed by integrating over travel time distributions between consecutive stops. However, the optimal route can not be fixed a-priori (as in standard TSP problems), as the vehicle also performs deliveries between stations and to the center, depending on the current needs. These deliveries enforce various constraints on the allowed routes. The overall probability is thus obtained by summing over all valid routes, given the known constraints.

Consider, for example, the case in which stations at A14, A1304, A43, and A5199 need to be visited before returning to the operations center. Figure 2a depicts this case, showing the portion of the network with at most two junctions away from A6. (The nodes to be visited are colored green.) The probability of beginning from the operations center at 8 a.m. and completing the route by 9 a.m., considering all possible paths, is:

$$\Pr(T < 3600) = 0.765,$$

computed using the approximate WMI module, where  $T$  is the overall time measured in seconds. Now suppose a constraint says that station A14 should be reached only after visiting A1304 (owing to a delivery request between these two stations). The probability then needs to be updated according to this evidence, which rules out part of the possible routes. Nonetheless, the probability of completing the task is almost unchanged:

$$\Pr(T < 3600 \mid t_{A14} > t_{A1304}) = 0.770,$$

<sup>9</sup><http://data.gov.uk/dataset/dft-eng-srn-routes-journey-times>

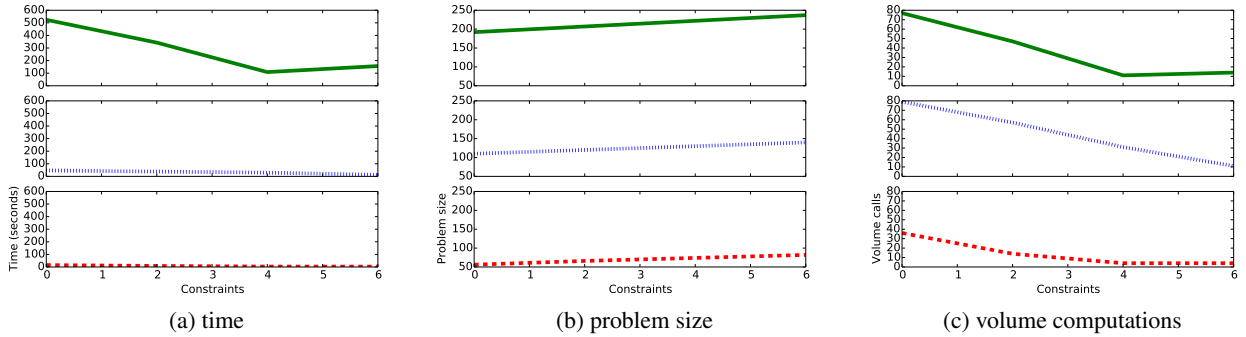


Figure 3: Probabilistic reasoning about the Strategic Road Network surrounding motorway A6. Figures 3a-3c plot cycle lengths of 5 (dotted red), 6 (finely dotted blue) and 7 (solid green), relating constraints to time, problem size and volume computations respectively.

where the minor increase is due to some slightly suboptimal routes being disallowed. Suppose further an additional constraint specifies that station A1304 should not be visited before 8:30 a.m. (say, because the package to deliver will not be available until then). This additional evidence brings success probability down to:

$$\Pr(T < 3600 \mid t_{A14} > t_{A1304} \wedge t_{A1304} \geq 1800) = 0.557.$$

Finally, suppose a last constraint were to require the station A5199 to be also visited after 8:30 a.m. (say, when a package to be delivered to the operations center will be made available). This additional constraint makes it infeasible to complete the route in the required time:

$$\Pr(T < 3600 \mid t_{A14} > t_{A1304} \wedge t_{A1304} \geq 1800 \wedge t_{A5199} \geq 1800) = 0.$$

Note that for such carefully constructed small-size problems in the 2-neighborhood case, exact and approximate procedures return the very same results in about the same time. (The exact procedure, however, quickly becomes infeasible for increasing cycle lengths.)

Using this example, which is to be seen as a cycle of length 5: A6–A14–A1304–A43–A5199–A6, as a template we randomly generated a number of test problems on the more complex 3-neighborhood setting, and plot an excerpt on extensive results on the dataset. These test problems are diverse in their modeling power: ranging from inequality constraints (e.g.,  $t_{A5199} \geq 1800$ ) to ordering constraints (e.g., A5199 after A1304), and Boolean combinations thereof, often leading to more than 300 complex SMT formulas. (Intuitively, if a SMT formula has  $n$  SMT literals, these can possibly denote joint piecewise potentials of  $n$  continuous random variables.) The figures depict the behaviors for cycles of length 5, 6 and 7, ordered by constraints against the time taken, the problem size (which is the number of complex SMT formulas in the theory), and the number of calls of  $\text{VOL}(\cdot, \cdot)$ . Besides demonstrating the

scalability of approximate WMI in such a setting, one also observes that while additional constraints increases the size of the theory, and thus, the number of random variables and volume computations, the time taken for conditional queries does not necessarily increase because suboptimal paths are eliminated (and so, marginalization is easier). To the best of our knowledge, a probabilistic inference system for hybrid specifications against intricate Boolean combinations of propositional and arithmetic constraints has not been deployed on such a scale previously.

## 5 CONCLUSIONS

We introduced a novel way to leverage a fast hashing-based approximate WMC methodology for inference with discrete and continuous random variables. On the one hand, SAT technology can now be exploited in challenging inference and learning tasks in hybrid domains. On the other, strong tolerance-confidence guarantees can be inherited in this more complex setting. WMI and weighted SMT theories allow a natural encoding of hybrid graphical networks while also admitting the specification of arithmetic constraints in conditional queries, all of which are difficult to realize in traditional representations. We demonstrated its practical efficacy in a complex novel application, and we believe, in general, the ideas of our approach would put hybrid domains within the reach of other WMC solvers.

### Acknowledgements

The authors thank Kuldeep Meel for helpful discussions. Vaishak Belle is partially funded by the Research Foundation-Flanders (FWO-Vlaanderen) project on Data Cleaning and the KU Leuven GOA on Declarative Modeling for Mining and Learning. Guy Van den Broeck is supported by the Research Foundation-Flanders.

## References

- Velleda Baldoni, Nicole Berline, Jesus De Loera, Matthias Köppe, and Michèle Vergne. How to integrate a polynomial over a simplex. *Mathematics of Computation*, 80(273):297–325, 2011.
- Clark Barrett, Roberto Sebastiani, Sanjit A. Seshia, and Cesare Tinelli. Satisfiability modulo theories. In Armin Biere, Marijn J. H. Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, chapter 26, pages 825–885. IOS Press, 2009.
- Mihir Bellare, Oded Goldreich, and Erez Petrank. Uniform generation of NP-witnesses using an NP-oracle. *Information and Computation*, 163(2):510 – 526, 2000.
- Vaishak Belle, Andrea Passerini, and Guy Van den Broeck. Probabilistic inference in hybrid domains by weighted model integration. In *IJCAI*, 2015.
- Supratik Chakraborty, Kuldeep S. Meel, and Moshe Y. Vardi. A scalable and nearly uniform generator of SAT witnesses. In *CAV*, pages 608–623, 2013.
- Supratik Chakraborty, Kuldeep S. Meel, and Moshe Y. Vardi. A scalable approximate model counter. In *CP*, pages 200–216, 2013.
- Supratik Chakraborty, Daniel J Fremont, Kuldeep S Meel, Sanjit A Seshia, and Moshe Y Vardi. Distribution-aware sampling and weighted model counting for SAT. *AAAI*, 2014.
- Mark Chavira and Adnan Darwiche. On probabilistic inference by weighted model counting. *Artificial Intelligence*, 172(6-7):772–799, April 2008.
- Mark Chavira, Adnan Darwiche, and Manfred Jaeger. Compiling relational Bayesian networks for exact inference. *International Journal of Approximate Reasoning*, 42(1-2):4–20, May 2006.
- Dmitry Chistikov, Rayna Dimitrova, and Rupak Majumdar. Approximate counting in SMT and value estimation for probabilistic programs. In *TACAS*, volume 9035 of *LNCS*, pages 320–334. Springer Berlin Heidelberg, 2015.
- Arthur Choi, Doga Kisa, and Adnan Darwiche. Compiling probabilistic graphical models using sentential decision diagrams. In *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, pages 121–132. Springer, 2013.
- Adnan Darwiche. New advances in compiling CNF to decomposable negation normal form. In *Proceedings of ECAI*, pages 328–332, 2004.
- Stefano Ermon, Carla P. Gomes, Ashish Sabharwal, and Bart Selman. Embed and project: Discrete sampling with universal hashing. In *NIPS*, pages 2085–2093, 2013.
- Stefano Ermon, Carla P. Gomes, Ashish Sabharwal, and Bart Selman. Taming the curse of dimensionality: Discrete integration by hashing and optimization. In *ICML*, pages 334–342, 2013.
- Stefano Ermon, Carla P. Gomes, Ashish Sabharwal, and Bart Selman. Low-density parity constraints for hashing-based discrete integration. In *ICML*, pages 271–279, 2014.
- Daan Fierens, Guy Van den Broeck, Joris Renkens, Dimitar Shterionov, Bernd Gutmann, Ingo Thon, Gerda Janssens, and Luc De Raedt. Inference and learning in probabilistic logic programs using weighted Boolean formulas. *Theory and Practice of Logic Programming*, 2013.
- Vibhav Gogate and Pedro Domingos. Probabilistic theorem proving. In *UAI*, pages 256–265, 2011.
- Carla P. Gomes, Ashish Sabharwal, and Bart Selman. Near-uniform sampling of combinatorial spaces using XOR constraints. In *NIPS*, pages 481–488, 2006.
- Carla P. Gomes, Ashish Sabharwal, and Bart Selman. Model counting. In Armin Biere, Marijn J. H. Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, chapter 20. IOS Press, 2009.
- Mark R Jerrum, Leslie G Valiant, and Vijay V Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theor. Comput. Sci.*, 43(2-3):169–188, 1986.
- Richard M. Karp, Michael Luby, and Neal Madras. Monte-carlo approximation algorithms for enumeration problems. *J. Algorithms*, 10(3):429–448, 1989.
- D. Koller and N. Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.
- Lukas Kroc, Ashish Sabharwal, and Bart Selman. Leveraging belief propagation, backtrack search, and statistics for model counting. *Annals OR*, 184(1):209–231, 2011.
- Steffen L Lauritzen and Frank Jensen. Stable local computation with conditional gaussian distributions. *Statistics and Computing*, 11(2):191–203, 2001.
- David J Lunn, Andrew Thomas, Nicky Best, and David Spiegelhalter. Winbugs – a Bayesian modelling framework: concepts, structure, and extensibility. *Statistics and computing*, 10(4):325–337, 2000.
- Christian Muise, Sheila A McIlraith, J Christopher Beck, and Eric I Hsu. Dsharp: fast d-DNNF compilation with sharpSAT. In *Advances in Artificial Intelligence*, pages 356–361. Springer, 2012.
- Kevin P Murphy. A variational approximation for Bayesian networks with discrete and continuous latent variables. In *UAI*, pages 457–466, 1999.
- Tian Sang, Paul Beame, and Henry A Kautz. Performing Bayesian inference by weighted model counting. In *AAAI*, volume 5, pages 475–481, 2005.
- Scott Sanner and Ehsan Abbasnejad. Symbolic variable elimination for discrete and continuous graphical models. In *AAAI*, 2012.
- Prakash P Shenoy and James C West. Inference in hybrid Bayesian networks using mixtures of polynomials. *International Journal of Approximate Reasoning*, 52(5):641–657, 2011.
- Michael Sipser. A complexity theoretic approach to randomness. In *STOC*, pages 330–335. ACM, 1983.
- Larry Stockmeyer. The complexity of approximate counting. In *STOC*, pages 118–126, New York, NY, USA, 1983. ACM.
- Dan Suciu, Dan Olteanu, Christopher Ré, and Christoph Koch. Probabilistic databases. *Synthesis Lectures on Data Management*, 3(2):1–180, 2011.
- Leslie G Valiant. The complexity of enumeration and reliability problems. *SIAM Journal on Computing*, 8(3):410–421, 1979.
- Guy Van den Broeck, Nima Taghipour, Wannes Meert, Jesse Davis, and Luc De Raedt. Lifted probabilistic inference by first-order knowledge compilation. In *IJCAI*, pages 2178–2185, 2011.
- Shenlong Wang, Alexander G. Schwing, and Raquel Urtasun. Efficient inference of continuous Markov random fields with polynomial potentials. In *NIPS*, pages 936–944, 2014.



---

# Bayesian Network Learning with Discrete Case-Control Data

---

**Giorgos Borboudakis**

Comp. Sci. Dept., University of Crete  
Institute of Computer Science, FORTH

**Ioannis Tsamardinos**

Comp. Sci. Dept., University of Crete  
Institute of Computer Science, FORTH

## Abstract

We address the problem of learning Bayesian networks from discrete, unmatched case-control data using specialized conditional independence tests. Those tests can also be used for learning other types of graphical models or for feature selection. We also propose a post-processing method that can be applied in conjunction with any Bayesian network learning algorithm. In simulations we show that our methods are able to deal with selection bias from case-control data.

## 1 INTRODUCTION

Most Bayesian network learning algorithms assume i.i.d. data. In many studies, such as **case-control** studies, this is not the case. In case-control studies data are selected based on one or multiple variables, usually using a comparable number of samples from different values of those variables, leading to non i.i.d. data. Such data are also called **artificially balanced** in the machine learning literature. Case-control sampling is often used in epidemiological or biomedical studies [Breslow, 1996], particularly when studying a disease that is relatively rare. Their goal is usually to identify differences between patients (cases) and healthy individuals (controls), such as identifying differentially expressed genes between the two groups from gene expression data. Case-control sampling is especially important when cases are very rare, as much fewer samples have to be collected compared to cross-sectional studies, significantly reducing the time and cost for obtaining the data.

We address the problem of learning Bayesian networks from discrete case-control data. This is challenging because case-control data do not necessarily represent a sample from the general population. In general, non i.i.d. sampling may lead to **selection bias**, which could alter the (conditional) independence relations in the data; case-

control sampling is a special case of such sampling. Because of that, one cannot usually use methods designed for i.i.d. data.

To the best of our knowledge, there has been only one previous approach to learn Bayesian networks from case-control data by Cooper [2000]. The idea is to use a Bayesian method to integrate over all possible values for all non-sampled cases and controls, assuming that those numbers are known a priori. This method is very impractical, mainly due to its high computational cost. There is a vast literature on methods for case-control data ([Rothman et al., 2008] contains a review of many methods, as well as relevant references) but they are mostly concerned with modeling the outcome (on which selection is based on), and thus are not applicable for modeling other measured variables or for Bayesian network learning.

Learning from case-control data is important for the following reasons. First, a lot of case-control datasets have accumulated over the years and such methods could be used to identify novel associations or possibly even causal relations. For instance, the NCBI GEO database contains thousands of biological datasets [Edgar et al., 2002, Barrett et al., 2013], many of which stem from case-control studies. In addition, this would allow co-analysis of data collected under different experimental designs. Currently, there exist several methods for learning causal networks from multiple heterogeneous datasets [Cooper and Yoo, 1999, Tillman and Spirtes, 2011, Hyttinen et al., 2013, Triantafillou and Tsamardinos, 2014]. Those methods are able to use observational and experimental i.i.d. data. Extending them for other types of data, such as case-control data, is a natural next step.

In this paper we define the problem and show the implications of case-control sampling on the observed independencies. We propose different conditional independence tests for discrete data, as well as a post-processing method that can be used with any Bayesian network learning algorithm. Finally, we investigate the behavior of our methods in various, simulated experiments and show that they are able to handle selection bias from case-control data.

## 2 PRELIMINARIES

We will briefly introduce the basic theory and notation used throughout the paper. Interested readers may refer to Pearl [2000] or Spirtes et al. [2000].

We use upper-case and lower-case letters to refer to random variables (e.g.  $X$ ) and values of those variables (e.g.  $x$ ), and bold letters to refer to sets of variables or values. Let  $\mathbf{V}$  be a set of random variables. A **Bayesian Network** (BN) over  $\mathbf{V}$  is a pair  $\mathcal{B} = \langle \mathcal{G}, \mathcal{P} \rangle$ , where  $\mathcal{G}$  is a **Directed Acyclic Graph** (DAG) representing conditional independencies between variables  $\mathbf{V}$ , and  $\mathcal{P}$  is the joint probability distribution of  $\mathbf{V}$ . We will use the terms variable and node interchangeably. The graph and distribution are connected through the **Markov Condition**: a variable is conditionally independent of all its non-descendants given its parents. The **skeleton**  $\mathcal{G}_S$  of a BN  $\mathcal{G}$  is the undirected graph which can be constructed by ignoring the orientations of  $\mathcal{G}$ . A triple of nodes  $\langle X, Y, Z \rangle$  is called a **collider** in  $\mathcal{G}$ , if  $X \rightarrow Y \leftarrow Z$  is in  $\mathcal{G}$ . Two variables  $X$  and  $Y$  are  **$d$ -separated** given a (possibly empty) set of variables  $\mathbf{Z}$  if and only if for all paths between  $X$  and  $Y$  one of the following is true: (a) there is a collider  $U \rightarrow V \leftarrow W$  on that path and neither  $V$  nor any of its descendants is in  $\mathbf{Z}$ , or (b) there is a consecutive triple  $\langle U, V, W \rangle$  that is not a collider and  $V$  is in  $\mathbf{Z}$ . If  $X$  and  $Y$  are not  $d$ -separated given  $\mathbf{Z}$  they are  **$d$ -connected**. We assume the **Faithfulness Condition** that (together with the Markov Condition) implies that *there is a  $d$ -connecting path between  $X$  and  $Y$  given  $\mathbf{Z}$ , if and only if  $X$  and  $Y$  are statistically dependent given  $\mathbf{Z}$* . We denote conditional dependence and independence of two variables  $X$  and  $Y$  given  $\mathbf{Z}$  as  $Dep(X; Y | \mathbf{Z})$  and  $Ind(X; Y | \mathbf{Z})$  respectively.

## 3 PROBLEM DEFINITION

In this work we consider discrete data from **unmatched case-control** studies. In unmatched studies samples are assumed to be sampled in an i.i.d. fashion from the respective subpopulations. We assume that the data have been selected based on a set of *measured* variables  $\mathbf{T}$ ; we will call those variables **selection variables**. In case  $\mathbf{T}$  contains only a single variable, we will refer to it as  $T$ . We denote with  $S$  a binary variable that indicates whether a sample has been selected or not. In our case, we assume that  $S$  only depends on  $\mathbf{T}$ ; their relation can be modeled by nodes with directed edges from each  $\mathbf{T}$  to  $S$ .

**Assumption 1.**  $S$  depends only on  $\mathbf{T}$  and all variables in  $\mathbf{T}$  have been measured.

Case-control sampling induces a type of **selection bias**. Selection bias arises if samples are less probably to be sampled based on some criteria. When analyzing such data it may happen that **spurious dependencies** are identified which do not exist in the general population, but are

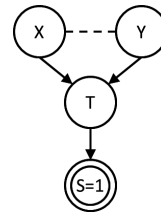


Figure 1: Conditioning on  $S = 1$  introduces a spurious dependence between  $X$  and  $Y$ .

due to the selection process. The reason for that is that the data  $\mathcal{D}$  are collected from the conditional distribution  $P(\mathcal{D} | S = 1)$ . Consider the example shown in Figure 1. Although  $X$  and  $Y$  are independent in the general population, a naive analysis that does not account for the sampling process would identify a spurious dependence between them. This happens because the data are selected conditional on  $S = 1$ ,  $d$ -connecting  $X$  and  $Y$  through  $T$ , as  $S$  is a descendant of  $T$  and  $T$  is a collider on the path  $X \rightarrow T \leftarrow Y$ .

The question is if and when it is possible to estimate the joint probability distribution of a set of variables  $\mathbf{X}$  in the general population,  $P(\mathbf{X})$ , from data collected with case-control sampling, that is following  $P(\mathbf{X} | S = 1)$ . Bareinboim et al. [2014] address the general problem of recoverability of conditional distributions when data are collected under selection. They show that the conditional distribution  $P(Y | X)$  is not recoverable when data are collected with case-control sampling. This result can be trivially extended to the case of estimating the joint distribution of a set of variables. Fortunately, they show that it is possible to recover the population distribution if the joint distribution of  $\mathbf{T}$  is known. Then,  $P(\mathbf{X})$  can be estimated as follows.

$$\begin{aligned} P(\mathbf{X}) &= \sum_{\mathbf{t}} P(\mathbf{X} | \mathbf{T} = \mathbf{t}) P(\mathbf{T} = \mathbf{t}) \\ &= \sum_{\mathbf{t}} P(\mathbf{X} | \mathbf{T} = \mathbf{t}, S = 1) P(\mathbf{T} = \mathbf{t}) \end{aligned} \quad (1)$$

The second equality follows by Assumption 1. The conditional probability  $P(\mathbf{X} | \mathbf{T} = \mathbf{t}, S = 1)$  can be directly estimated from  $\mathcal{D}$ . Thus, in order to estimate  $P(\mathbf{X})$  for any set of variables we only need the joint probability distribution of  $\mathbf{T}$  in the general population. This could either be provided as prior knowledge by a domain expert or from the literature, or estimated from an external data source.

**Assumption 2.** The joint probability distribution of  $\mathbf{T}$  in the general population is known.

Notice that there are cases where the equality  $P(\mathbf{X}) = P(\mathbf{X} | S = 1)$  holds. For example, if none of the variables in  $\mathbf{X}$  is dependent with  $\mathbf{T}$ ,  $P(\mathbf{X})$  can be directly estimated from  $\mathcal{D}$ . We will further investigate the conditions under which this holds in the next section.

We conclude with some comments on our assumptions. Assumption 1 is reasonable and probably holds for most case-control studies. In case it is violated additional spurious dependencies may be introduced. Regarding Assumption 2: although it may be restrictive in some cases, prior information about the joint distribution is often available. In fact, many methods for analyzing case-control data require such prior information. For example, in logistic regression models for an outcome that has been selected on (e.g. disease), such knowledge is necessary in order to estimate the intercept of the model, although it is not needed in order to estimate the remaining parameters [Breslow and Day, 1980].

## 4 IMPLICATIONS OF CONDITIONING ON S

In the previous section we saw an example where conditioning on  $S = 1$  introduces a spurious dependence. Next, we will further investigate how the dependencies and independencies are affected after conditioning on  $S = 1$ . Most of those results are based on previous results by Spirtes et al. [2000] (see Section 9.3). Those results are general, allowing for  $S$  to be in any position in the graph. We will show their consequences for the special case of case-control sampling.

First we investigate whether conditioning on  $S = 1$  removes any dependencies that exist in the general population. Spirtes et al. [2000] have characterized all situations where this happens.

**Corollary 1** (Adapted from [Spirtes et al., 2000]). *If  $Dep(X; Y|\mathbf{Z})$ , then  $Ind(X; Y|\mathbf{Z}, S = 1)$  holds if and only if there exists a path  $U$  between  $X$  and  $Y$  such that (a) every collider on  $U$  has a descendant in  $\mathbf{Z}$ , (b) no non-collider in  $U$  is in  $\mathbf{Z}$ , and (c)  $S$  is a non-collider on every such path.*

In our case there is no such path because the third condition can never be satisfied, as  $S$  does not have any outgoing edges. Because of that,  $S$  can only be a collider on all such paths. Therefore, *conditioning on  $S = 1$  does not remove any dependencies.*

The next corollary characterizes the cases where a conditional independence may turn into a dependence.

**Corollary 2** (Adapted from [Spirtes et al., 2000]). *If  $Ind(X; Y|\mathbf{Z})$ , then  $Dep(X; Y|\mathbf{Z}, S = 1)$  holds if and only if there exists a path  $U$  between  $X$  and  $Y$  such that (a) no non-collider on  $U$  is in  $\mathbf{Z} \cup \{S\}$ , (b) every collider on  $U$  has a descendant in  $\mathbf{Z} \cup \{S\}$ , and (c) some collider on  $U$  does not have a descendant in  $\mathbf{Z}$ .*

Based on this result, we will characterize all cases where a spurious dependence will appear in the graph after conditioning on  $S = 1$  that cannot be removed by conditioning

on any set of variables. We proceed by stating and proving the result.

**Theorem 1.** *Let  $X, Y$  be two variables. If  $\exists \mathbf{Z} Ind(X; Y|\mathbf{Z})$ , then  $\forall \mathbf{Z}' Dep(X; Y|\mathbf{Z}', S = 1)$  holds if and only if there is a node  $W$  such that (a)  $X \rightarrow W \leftarrow Y$ , and (b)  $W = S$  or  $W$  is an ancestor of  $S$ .*

*Proof.*

**Sufficiency:** Follows trivially, as conditioning on  $S$  d-connects  $X$  and  $Y$  through  $W$ .

**Necessity:** We will show this by contradiction. Assume that there is no  $W$  satisfying both conditions. From Corollary 2 we know that for some  $\mathbf{Z}$  satisfying both  $Ind(X; Y|\mathbf{Z})$  and  $Dep(X; Y|\mathbf{Z}, S = 1)$ , there is a path  $U$  between  $X$  and  $Y$  with at least one collider  $V$  on  $U$  that is also an ancestor of  $S$ . The only case where this holds is if there is at least one node between  $X$  and  $V$  or  $Y$  and  $V$  on  $U$ . But then at least one of those nodes has to be a non-collider and we could d-separate  $X$  and  $Y$  by conditioning on any such noncollider, contradicting our assumptions.  $\square$

In words, this theorem characterizes all cases where two variables  $X$  and  $Y$  can be d-separated in the population graph, but cannot d-separated by any set after conditioning on  $S$ . We will later use this to learn Bayesian networks from case-control data.

## 5 CONDITIONAL INDEPENDENCE TESTING

As we saw in the previous section, conditioning on  $S$  may introduce spurious dependencies in the data. Because of that, one cannot use independence tests designed for i.i.d. data. In this section we will describe various conditional independence tests for case-control data.

### 5.1 TEST STATISTIC

As a test statistic we consider the conditional mutual information (CMI)  $I(X; Y|\mathbf{Z})$ , which is defined as:

$$I(X; Y|\mathbf{Z}) = \sum_{x, y, \mathbf{z}} P(x, y, \mathbf{z}) \log \frac{P(x, y, \mathbf{z})P(\mathbf{z})}{P(x, \mathbf{z})P(y, \mathbf{z})} \quad (2)$$

Assuming that we know the distribution of  $\mathbf{T}$ , we can use Equation 1 to compute  $I(X; Y|\mathbf{Z})$  for any variables  $X, Y$  and  $\mathbf{Z}$ . For the case of i.i.d. data the CMI is closely related to the  $G$ -statistic used by the  $G$ -test:

$$G(X; Y|\mathbf{Z}) = 2 \cdot N \cdot I(X; Y|\mathbf{Z}) \quad (3)$$

where  $N$  is the sample size. Under the null hypothesis this statistic is asymptotically  $\chi^2$  distributed with  $(|X| - 1)(|Y| - 1)|Z|$  degrees of freedom.

Unfortunately, the  $G$ -test cannot be trivially applied to case-control data. Intuitively, the reason is that  $N$  case-control samples are not always equivalent to  $N$  samples from an i.i.d. dataset. We will show the intuition behind this with an example <sup>1</sup>.

**Example 1.** Assume that  $\mathbf{T}$  contains a single binary variable  $T$  and we sample  $N = 1000$  samples, 500 for each value of  $T$ , and that  $P(T = 0) = 0.2$ . The probability of  $\mathbf{X}$  given by Equation 1 is  $P(\mathbf{X}) = P(\mathbf{X}|T = 0) \cdot 0.2 + P(\mathbf{X}|T = 1) \cdot 0.8$ . If we use  $N$  as our sample size, we essentially assume that we have estimated  $P(\mathbf{X}|T = 0)$  and  $P(\mathbf{X}|T = 1)$  using 200 and 800 samples respectively even though we used 500 for each of them. As a result, we **overestimate and underestimate the variance in the estimation of  $P(\mathbf{X}|T = 1)$  and  $P(\mathbf{X}|T = 0)$  respectively, which can lead to false results.**

Next we consider various strategies to deal with this.

## 5.2 UNDERSAMPLING

The trivial approach is to use a subset of the samples such that the proportion of values of  $\mathbf{T}$  in the resulting dataset coincides with the distribution of  $\mathbf{T}$ . For Example 1 we could use 125 samples with  $T = 0$  and 500 samples with  $T = 1$ , as  $125/625 = 0.2$  and  $500/625 = 0.8$ , and perform a standard independence test. In general, one can use at most  $N = \min_{\mathbf{t}} N(\mathbf{T} = \mathbf{t}) / P(\mathbf{T} = \mathbf{t})$  samples, where  $N(\mathbf{T} = \mathbf{t})$  is the number of samples with  $\mathbf{T} = \mathbf{t}$  (proof omitted).

There are several downsides to this approach. First, undersampling often ignores a significant amount of samples (375 in the previous example), possibly reducing the power of the test. Second, the result may vary a lot, depending on the selected samples. One possibility to reduce this variance is to create multiple datasets by undersampling, perform a test on each such dataset and combine the results somehow (e.g. taking the median p-value). Finally, undersampling may be problematic if the marginal distribution of  $T$  contains extreme values. In the previous example, if  $P(\mathbf{X}|T = 0)$  was 0.01, only 5 samples with  $T = 0$  could be used to (approximately) satisfy the marginals. In practice, those values will often be even more extreme.

## 5.3 A PERMUTATION TEST

Permutation tests are non-parametric procedures for statistical significance testing. The basic idea is that, under the

<sup>1</sup>We have also conducted several, anecdotal simulations which confirm this problem.

null hypothesis, one can permute the data in an appropriate way to generate another, permuted dataset. *Specifically, for a permutation test to be exact, the permutation has to be performed in a way that preserves the distribution of the observations under the null hypothesis* [Good, 2004]. Then, the test statistic computed on that dataset is a sample from its null distribution. Because the number of all permutations is usually astronomically large, making complete enumeration infeasible, one usually resorts to Monte Carlo approximations that sample a relatively small number of permutations (usually between 1000 and 10000). The  $p$ -value is computed as the proportion of permutation statistics that are at least as extreme as the statistic on the original data.

**Permutation Testing for Discrete Data.** For conditional independence testing the null hypothesis is that  $X$  and  $Y$  are conditionally independent given  $\mathbf{Z}$ . This means that conditional independence holds for any value  $\mathbf{z}$  of  $\mathbf{Z}$ . A permuted dataset can be created by randomly permuting the columns of  $X$  and  $Y$  for each value  $\mathbf{z}$  of  $\mathbf{Z}$  [Tsamardinos and Borboudakis, 2010]. For example, if  $Z$  is a binary variable, a permuted dataset is created by splitting the original dataset  $\mathcal{D}$  into two datasets,  $\mathcal{D}_{Z=0}$  and  $\mathcal{D}_{Z=1}$ , randomly permuting the columns of  $X$  and  $Y$  on each of them and then combining the resulting datasets. This results in an exact test, as the conditional distribution of  $X$  and  $Y$  for each value of  $\mathbf{Z}$  remains fixed.

We use the same procedure for discrete case-control data using the CMI as our test statistic. It is important to note that *this permutation approach does not always preserve the distribution of  $X$  and  $Y$  under the null* when applied to case-control data. We show this with the following example. Let  $X$  be a discrete variable,  $T$  a binary variable and  $P(T = 0) = 0.2$ . Now, assume that for some value  $x$  of  $X$  we have 100 samples with  $T = 0$  and 50 samples with  $T = 1$  in the original dataset, and 50 and 100 respectively for some permuted dataset. Then,  $P(x) = 100/500 \cdot 0.2 + 50/500 \cdot 0.8 = 0.12$  for the original dataset but  $P(x) = 50/500 \cdot 0.2 + 100/500 \cdot 0.8 = 0.18$  for the permuted dataset. Let  $Y$  be another discrete variable with the same marginals as  $X$ . Then, the joint distribution of  $X$  and  $Y$  under the null is  $P(x, y) = 0.12 \cdot 0.12$  in the original dataset and  $P(x, y) = 0.18 \cdot 0.18$  in the permuted dataset. Thus, their joint distribution under the null is not invariant for this type of permutations.

We conducted various simulations to investigate the behavior of this test and, although this approach does not result in an exact test, they suggest that it works reasonably well in practice; the results are presented in Section 7.

## 5.4 AN ASYMPTOTIC TEST

An interesting observation that we made is that, under the null hypothesis, the permutation distribution of the  $G$ -

---

**Algorithm 1** Estimate Effective Sample Size

---

**Input:**  $P(\mathbf{T}), N(\mathbf{T}), N, K$ **Output:**  $N_{ESS}$ 

- 1: **for**  $i \leftarrow 1 : K$  **do**
  - 2:    $X \leftarrow \text{Random}(\text{Uniform}, \text{Binary}, N)$
  - 3:    $Y \leftarrow \text{Random}(\text{Uniform}, \text{Binary}, N)$
  - 4:    $Stats_i \leftarrow \text{MutualInformation}(X, Y, N(\mathbf{T}), P(\mathbf{T}))$
  - 5: **end for**
  - 6:  $Stats \leftarrow \text{Sort}(Stats, \text{Ascending})$
  - 7:  $Stats' \leftarrow \text{Inverse-}\chi^2\text{-Cdf}(0.1/(K-1); 1, \text{DoF} = 1)$
  - 8:  $N_{ESS} \leftarrow \text{Median}(1/2 \cdot Stats' / Stats)$
  - 9:  $N_{ESS} \leftarrow \text{Min}(N_{ESS}, N)$
- 

statistic computed on the case-control data, for some unknown number of samples  $N$ , seems to also follow a  $\chi^2$  distribution with  $(|X| - 1)(|Y| - 1)|\mathbf{Z}|$  degrees of freedom. We observed this behavior for a large number of different: (i) conditional and unconditional tests, (ii) probability distributions of  $\mathbf{T}$ , and (iii) types of discrete variables  $X$ ,  $Y$  and  $\mathbf{Z}$ . We conjecture that this is always the case.

**Conjecture 1.** *The  $G$ -statistic  $G(X; Y|\mathbf{Z})$  as defined by Equation 3 and computed for case-control data using Equations 1 and 2 is asymptotically distributed as a  $\chi^2$  random variable with  $(|X| - 1)(|Y| - 1)|\mathbf{Z}|$  degrees of freedom for some unknown number of samples  $N$ .*

We will call this unknown number of samples the **effective sample size** and denote it as  $N_{ESS}$ . Based on this conjecture, we will devise a simple procedure to estimate  $N_{ESS}$ .

Let  $\mathcal{D}$  be a dataset obtained from case-control sampling,  $P(\mathbf{T})$  be the joint distribution of  $\mathbf{T}$ ,  $N(\mathbf{T})$  be the number of samples in  $\mathcal{D}$  for each value of  $\mathbf{T}$ , and  $N$  be the total number of samples in  $\mathcal{D}$ . Suppose that we generate a large number  $K$  of independent random variables  $X$  and  $Y$ , each of  $N$  samples, assuming that the first  $N(\mathbf{T} = \mathbf{t}_0)$  samples correspond to  $\mathbf{T} = \mathbf{t}_0$ , the next  $N(\mathbf{T} = \mathbf{t}_1)$  to  $\mathbf{T} = \mathbf{t}_1$  and so on, and then compute their mutual information. Let  $Stats$  contain all statistics in ascending order. If  $K$  is large enough we would expect the  $i$ -th value in  $Stats$ ,  $Stats_i$ , to correspond to a  $p$ -value of  $i/(K - 1)$ . According to Conjecture 1 the  $G$ -statistic for some  $N_{ESS}$  of any two independent random variables follows a  $\chi^2$  distribution. Thus, for each such  $p$ -value, we can use the inverse  $\chi^2$  cumulative distribution to compute its corresponding statistic  $Stats'$ . We know that  $Stats_i = I(X_i; Y_i)$  and that  $Stats'_i \simeq 2 \cdot N_{ESS} \cdot I(X_i; Y_i)$ . Thus, we can estimate  $N_{ESS}$  as  $N_{ESS} \simeq 1/2 \cdot Stats'_i / Stats_i$ . Because the procedure is not exact, we suggest to compute this value for each pair of  $Stats$  and  $Stats'$  values, and use the median value as an estimate for  $N_{ESS}$ . Naturally, this value cannot be larger than  $N$ , so we use the minimum of those values. The procedure is shown in Algorithm 1.

The method only needs to be applied once before analyzing

a dataset, adding only a constant computational overhead. As a result, the cost of analyzing a case-control dataset is essentially identical to analyzing any other dataset, up to a constant additive factor.

## 6 BAYESIAN NETWORK LEARNING

We propose two different strategies for learning Bayesian networks from case-control data.

One is to use a test suited for case-control data with any existing constraint-based method. This strategy also allows one to learn other graphs such as Maximal Ancestral Graphs [Richardson and Spirtes, 2002], or to perform feature selection using a conditional independence based method [Tsamardinos et al., 2006].

Another approach is to learn a network using an independence test suited for i.i.d. data and perform a **post-processing** step to correct the graph by identifying and removing spurious dependencies using an independence test for case-control data. Theorem 1 characterizes all cases where a spurious dependence will be identified. Of course, we cannot directly apply Theorem 1 as we do not know the real DAG. Instead, we will use the skeleton of the DAG without any orientations. The next corollary characterizes all potentially spurious edges in a skeleton.

**Corollary 3.** *Let  $G_S$  be the skeleton of a DAG  $\mathcal{G}$ . An edge between variables  $X$  and  $Y$  is **potentially spurious**, if and only if there is a node  $W$  such that (a)  $X, Y$  and  $W$  are adjacent and form a triangle, and (b)  $W = S$  or there is a potentially directed path from  $W$  to  $S$  (the path cannot go through  $X$  or  $Y$ ). As  $S$  will not be in  $\mathcal{G}$  we have to check if  $W \in \mathbf{T}$  or if  $W$  is a potential ancestor of any variable in  $\mathbf{T}$ .*

This result follows from Theorem 1 and is stated without proof. This directly suggests how to use it with existing learning algorithms. After identifying  $\mathcal{G}$ , take its skeleton  $G_S$ , check for triples  $X, Y, W$  that satisfy those criteria, and finally try to remove potentially spurious edges by performing a series of independence tests with an appropriate method. Note that  $W$  never has to be conditioned on in those tests as it either is a collider and would  $d$ -connect  $X$  and  $Y$  or, if not, the edge between  $X$  and  $Y$  can not be spurious and should not be removed. The second condition can be checked by removing all edges at  $X$  and  $Y$  and checking whether  $W$  and  $\mathbf{T}$  are connected by a path.

We have to point out that this approach may not be optimal. Instead of the skeleton, there may be a way to partially orient the graph and further narrow down the cases where Corollary 3 applies. For example, if the edge from  $W$  to  $X$  is oriented towards  $X$ , then the edge between  $X$  and  $Y$  cannot be due to a spurious dependence, but applying Corollary 3 on the skeleton will try to remove it. However, this is not trivial; a naive application of the PC rules may

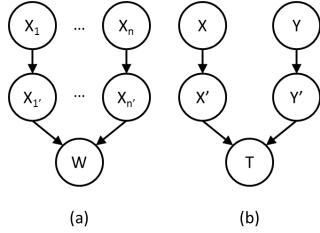


Figure 2: (a) Graphical representation of noisy model. (b) Collider model for the first set of experiments.

result in false orientations due to the presence of spurious edges. We did not further investigate this possibility.

## 7 EXPERIMENTAL EVALUATION

We performed simulations to investigate the behavior of the proposed independence tests in different situations. We consider only a single selection variable  $T$ . When we generate data, we select an equal number of samples for each value of  $T$  (that is,  $T$  is uniformly distributed in the case-control data). For a given Bayesian network we compute the marginal distribution of  $T$  using an exact inference algorithm implemented in the Bayes Net toolbox [Murphy, 2001]. We used  $K = 100000$  to estimate the effective sample size.

First, we evaluate the tests in simple Noisy-MAX and Noisy-SUM scenarios. Then, we investigate the sensitivity of the tests with respect to the prior distribution of  $T$ . Finally, we compare the proposed Bayesian network learning strategies on the INSURANCE network [Binder et al., 1997]. We used MATLAB to conduct the simulations and create the figures (the code is available at <http://www.mensxmachina.org/>).

### 7.1 NOISY-MAX AND NOISY-SUM

We use the model proposed by [Srinivas, 1993], a generalization of the Noisy-OR model [Pearl, 1988], to generate data from a Noisy-MAX or Noisy-SUM distribution. Those models are members of the family of independence of causal influences (ICI) models [Heckerman and Breese, 1994]. A graphical representation of noisy models is shown in Figure 2. The nodes  $X'_i$  are called **inhibitor** nodes and are used to introduce noise in the model. Noisy-MAX distributions in particular are interesting as they have been shown to be very common in practice [Zagorecki and Druzdzel, 2006].

#### 7.1.1 Setup

In all our experiments the inhibitor nodes take the same number of values as their parents and their distribution is:  $P(X'_i = 0 | X_i = 0) = 1$ ,  $P(X'_i = k | X_i = k) = 1 - e$  and

$P(X'_i = k | X_i \neq k) = e / (|X_i| - 1)$ , where  $e$  is the noise parameter. The value of  $W$  is a deterministic function of its parent values (MAX or SUM).

We use two different cases for our evaluation. The first is a simple collider graph (see Figure 2 (b)). Here we test whether  $X$  and  $Y$  are unconditionally independent. We use this to evaluate the ability of the tests to handle the case of spurious dependencies. The second is a chain graph ( $X \rightarrow X' \rightarrow T \rightarrow T' \rightarrow Y$ ), with two additional nodes – inhibitor node pairs, one into  $T$  and one into  $Y$ . For this case we test whether  $X$  and  $Y$  are unconditionally and conditionally independent given  $T$ . This is done to investigate the behavior of the tests in case no spurious dependencies are present.

For the collider case we generated data from the Noisy-MAX and Noisy-SUM distributions, and for the chain graph we generated data from the Noisy-MAX distribution. The parameters we used are: noise  $e \in \{0, 0.3, 0.7\}$ , sample size  $N \in \{250, 1000\}$ , range of values  $r \in \{2, 4\}$  for  $X$  and  $Y$ . We used 6 independence tests:  $G^2$  test, Permutation  $G^2$  test,  $G^2_{cc}$  test for case-control data, Permutation  $G^2_{cc}$  test for case-control data, Undersampling  $G^2_u$  test, Bootstrapping and Undersampling  $G^2_u$  test using the median  $p$ -value. For the permutation tests we used 1000 permutations, and for the bootstrapping test we used 500 samples.

#### 7.1.2 Results

The results for the collider and chain graphs are shown in Figures 3 and 4. In each figure we show the empirical CDF function of the  $p$ -values. In case independence holds, the  $p$ -values should be uniformly distributed and the CDF should be on the diagonal. In case of dependence, we would ideally have low  $p$ -values only. We use  $\text{Test}(X; Y)$  and  $\text{Test}(X; Y | T)$  to refer to the unconditional and conditional tests of  $X$  and  $Y$ . The first and last two columns of each group of figures correspond to data with  $r = 2$  and  $r = 4$  respectively.

**$G^2$  and Permutation  $G^2$ .** For the collider graph both tests identify a spurious dependence, as expected, unless the noise is too high or the sample size is too low. For the chain graph, where case-control sampling does not affect the independencies, both tests perform well. The asymptotic test does not always produce calibrated  $p$ -values for the test  $\text{Test}(X; Y | T)$ , agreeing with previous results [Tsamardinos and Borboudakis, 2010]. The simulations confirm that tests designed for i.i.d. data should not be applied on case-control data.

**$G^2_{cc}$  and Permutation  $G^2_{cc}$**  For the collider graph, both tests produce  $p$ -values close to the ideal uniform distribution (black diagonal line), or overestimate the  $p$ -value; this can be seen especially in the noiseless Noisy-SUM case. Although this is not ideal, it is still useful, as the significance level upper bounds the actual type I error. Unfortu-

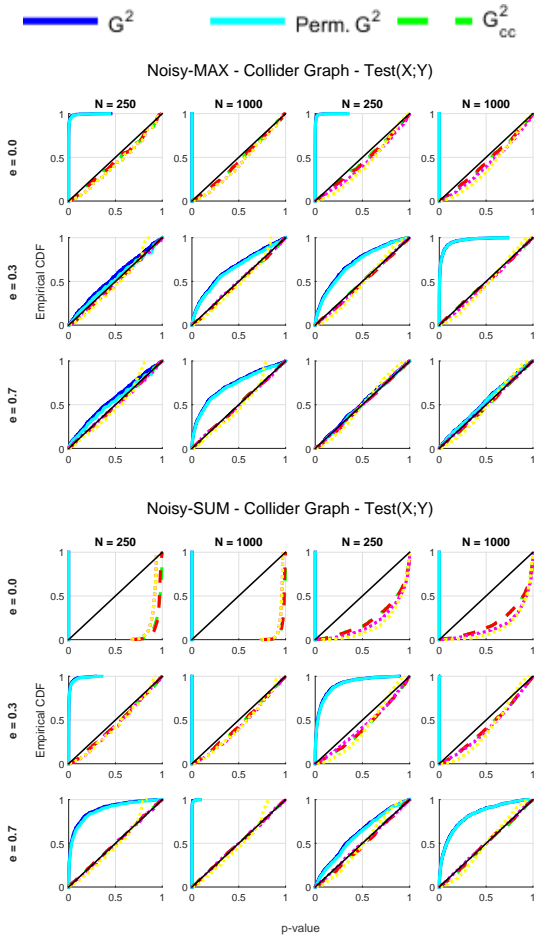


Figure 3: Results for the collider graph.

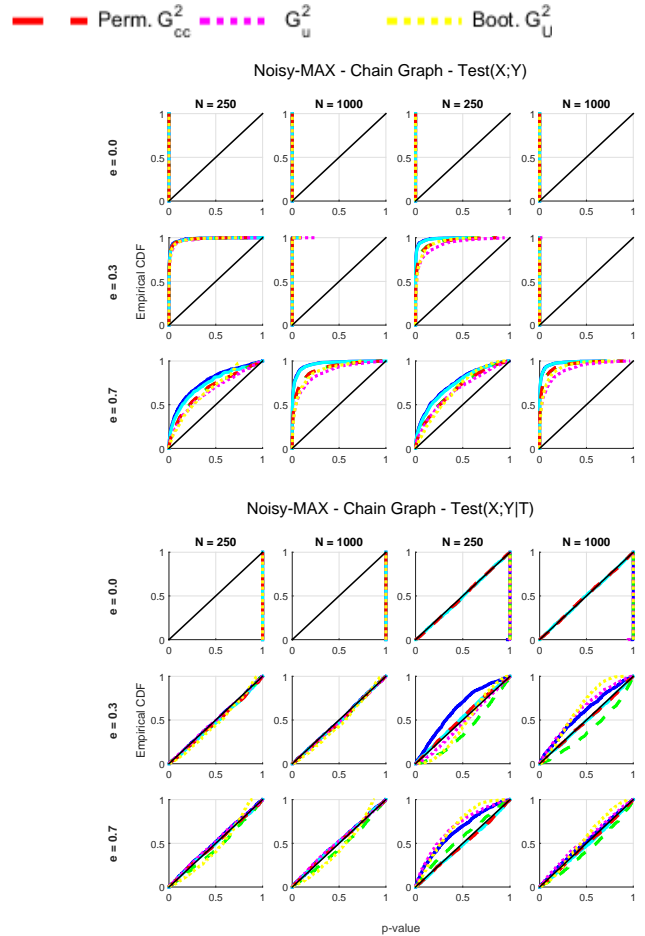


Figure 4: Results for the chain graph.

nately, we were not able to identify the circumstances under which this happens. For the chain graph, both tests perform reasonably well. In comparison to the  $G^2$  and Permutation  $G^2$  tests, the specialized tests have less power. Nevertheless, they will still be useful for network learning using the post-processing method, as it combines the best of both worlds. Again, the permutation test produces calibrated  $p$ -values for  $\text{Test}(X; Y|T)$ , whereas the asymptotic one does not.

**$G^2_u$  and Bootstrapping  $G^2_u$ .** Again, both tests perform similarly to the other specialized tests. However, undersampling exhibits a large variance, as it highly depends on the selected samples. The bootstrapping version reduces this variance, but its distribution has a heavy tail close to one. This bias may be the result of taking the median  $p$ -value. We discourage the use of the bootstrapping method, but there may be other similar approaches that do not exhibit this behavior.

**Comparison of  $G^2_{cc}$  and Permutation  $G^2_{cc}$**  Figure 5 (a,b) shows that the  $G^2_{cc}$  and Permutation  $G^2_{cc}$  produce almost identical  $p$ -values on the unconditional tests. In Figure 5

(c,d) we compare the asymptotic and permutation tests for the conditional case with noisy data and  $r = 4$  (Figure 4, bottom right). We see that in this case the  $G^2_{cc}$  does not produce the same  $p$ -values as the permutation  $G^2_{cc}$  test. However, the same behavior can be observed for the standard  $G^2$  test and the permutation  $G^2$  test. Also, for both cases, the  $p$ -values of the asymptotic tests are highly correlated with the ones of the permutation tests. The results suggest that the  $G^2_{cc}$  test is a reasonable approximation to the permutation version.

## 7.2 SENSITIVITY TO $P(T)$

We conducted a small experiment to investigate the sensitivity of the tests to the distribution of  $T$ . We simulated 1000 datasets from a Noisy-MAX collider graph with binary variables  $X$  and  $Y$ , with  $e = 0$  and  $N \in \{250, 1000\}$ . The distribution of  $T$  is  $P(T = 0) = 0.25$  and  $P(T = 1) = 0.75$  and we selected 500 samples for each value of  $T$ . In order to measure the sensitivity of the methods to the specified marginal distribution we computed the area under the empirical CDF of the  $p$ -values. Values close to 0.5

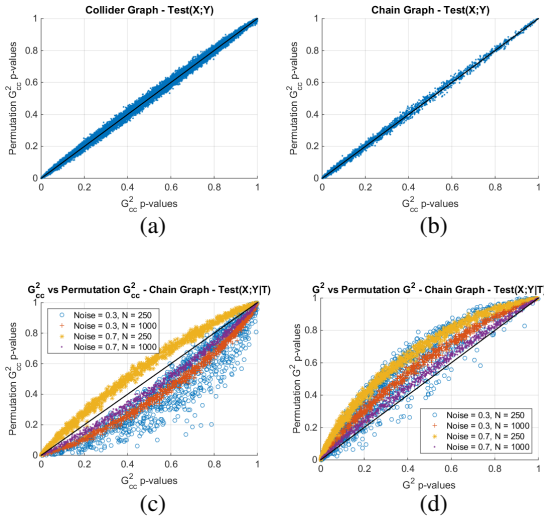


Figure 5: Comparison of the  $p$ -values from the  $G^2_{cc}$  and Permutation  $G^2_{cc}$  tests (a) on all tests for the collider graph (b) on the unconditional tests for the chain graph. Comparison on noisy data with  $r = 4$  of (c)  $G^2_{cc}$  vs Permutation  $G^2_{cc}$  (d)  $G^2$  vs Permutation  $G^2$ .

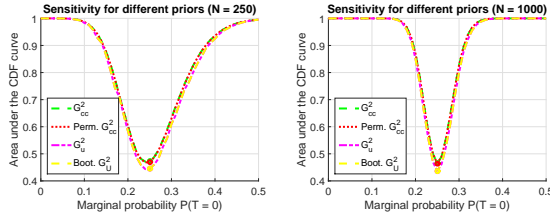


Figure 6: Sensitivity of methods to prior distribution.

indicate that the  $p$ -values are uniformly distributed (this is not always true, but should be reasonable in our case due to the convexity/concavity and monotonicity of the CDF; see Figure 3 for  $e = 0$  and  $N = 1000$ ). The results are summarized in Figure 6. We only show results from  $P(T = 0) = 0$  to  $0.5$ . We see that the methods are sensitive to the correct specification of the prior distribution, and that their sensitivity highly depends on the sample size. For  $N = 250$ , small deviations are acceptable, whereas for  $N = 1000$  even deviations of  $0.05$  significantly reduce the ability of the tests to detect spurious dependencies. Of course, this can not be generalized and it may highly vary for different distributions, but it indicates that those methods have to be used with care.

### 7.3 INSURANCE NETWORK

We evaluated our methods on the INSURANCE network [Binder et al., 1997]. It contains 27 nodes and 52 edges. This network is appropriate for our purposes as it has many nodes for introducing spurious dependencies that also have

Table 1: Characteristics of the selection variables. The second row shows the number of spurious dependencies induced by selecting on those variables. The last four rows show their marginal distribution.

| Node       | 18    | 19    | 20      | 21    | 25    | 26    |
|------------|-------|-------|---------|-------|-------|-------|
| Spurious   | 6     | 13    | 6       | 14    | 9     | 5     |
| $P(T = 0)$ | 0.001 | 0.788 | 0.844   | 0.541 | 0.888 | 0.965 |
| $P(T = 1)$ | 0.999 | 0.09  | 0.08    | 0.286 | 0.054 | 0.018 |
| $P(T = 2)$ | -     | 0.09  | 0.077   | 0.12  | 0.036 | 0.011 |
| $P(T = 3)$ | -     | 0.032 | 1.2e-05 | 0.052 | 0.023 | 0.007 |

relatively extreme distributions.

#### 7.3.1 Setup

We selected 6 nodes from the INSURANCE network as selection variables. The selection variables as well as their characteristics are shown in Table 1.

**Algorithms.** For Bayesian network learning we used the PC algorithm with Heuristic 3, as described in [Spirtes et al., 2000] (Section 5.4.2), except with an additional modification that sets an upper limit on the size of the conditioning set for each test. This is necessary especially for lower sample sizes, as conditioning on many variables tends to give very high  $p$ -values. In our simulations we set that parameter to 3 (maximum in-degree in the network). The significance level was set to  $0.05$  for all tests. We used 5 different methods to learn the network from case-control data: (a)  $G^2$  test, (b)  $G^2_{cc}$  test for case-control data, (c) Undersampling  $G^2_u$  test, (d)  $G^2$  test + post-processing with  $G^2_{cc}$  and (e)  $G^2$  test + post-processing with  $G^2_u$ . Methods (a-c) did not apply the post-processing step. Whenever the  $G^2_u$  test was used, undersampling was performed only once for each dataset. In addition, we also ran the PC algorithm with the  $G^2$  test on i.i.d. data to compare our methods against (Reference). The reference results should be close to the best achievable performance for a given sample size.

**Data.** Again, we generated data with equal proportions of each value of  $T$ . For each selection variable, as well as for the reference case, we generated 100 datasets for each of three different sample sizes  $N \in \{1000, 10000, 100000\}$ .

#### 7.3.2 Results

The results are summarized in Table 2. For each method we report the extra edges (“+”) and missing edges (“-”), averaged over all 100 runs.

**PC with  $G^2$  test.** We observe that ignoring the sampling and using PC with the standard  $G^2$  test performs very well for  $N = 1000$  and does not identify many spurious edges. However, as expected, it identifies a significant amount of extra edges with larger sample sizes.

**PC with  $G^2_{cc}$  and  $G^2_u$  tests.** The case-control tests with



Table 2: Results on the INSURANCE network. Extra edges are denoted with “+” and missing edges with “-”.

| Method          | T = 18           |      | T = 19 |      | T = 20 |      | T = 21 |      | T = 25 |      | T = 26 |      | Reference |      |       |
|-----------------|------------------|------|--------|------|--------|------|--------|------|--------|------|--------|------|-----------|------|-------|
|                 | +                | -    | +      | -    | +      | -    | +      | -    | +      | -    | +      | -    | +         | -    |       |
| <b>N = 1K</b>   | $G^2$            | 1.00 | 26.83  | 0.10 | 29.10  | 0.22 | 28.81  | 0.15 | 29.54  | 0.49 | 28.59  | 0.10 | 28.91     | 0.14 | 28.73 |
|                 | $G_{cc}^2$       | 0.35 | 32.19  | 0.28 | 33.51  | 0.23 | 34.20  | 0.36 | 30.97  | 0.21 | 34.44  | 0.31 | 35.11     |      |       |
|                 | $G_u^2$          | 0.33 | 32.15  | 0.28 | 34.15  | 0.23 | 34.41  | 0.21 | 32.51  | 0.32 | 34.82  | 0.30 | 34.94     |      |       |
|                 | $G^2 + G_{cc}^2$ | 1.00 | 26.84  | 0.09 | 29.10  | 0.22 | 28.88  | 0.15 | 29.54  | 0.49 | 28.65  | 0.10 | 28.94     |      |       |
|                 | $G^2 + G_u^2$    | 1.00 | 26.84  | 0.09 | 29.14  | 0.22 | 28.92  | 0.15 | 29.54  | 0.49 | 28.66  | 0.10 | 28.94     |      |       |
| <b>N = 10K</b>  | $G^2$            | 6.44 | 11.24  | 3.04 | 14.33  | 1.84 | 14.15  | 0.85 | 14.54  | 3.70 | 14.70  | 1.61 | 13.65     | 0.23 | 14.35 |
|                 | $G_{cc}^2$       | 0.00 | 16.96  | 0.00 | 18.89  | 0.00 | 19.81  | 0.00 | 16.38  | 0.00 | 20.82  | 0.01 | 21.90     |      |       |
|                 | $G_u^2$          | 0.00 | 16.93  | 0.00 | 20.20  | 0.02 | 20.66  | 0.00 | 17.19  | 0.10 | 21.18  | 0.00 | 22.08     |      |       |
|                 | $G^2 + G_{cc}^2$ | 0.50 | 13.07  | 0.18 | 16.51  | 0.00 | 16.72  | 0.00 | 16.32  | 1.09 | 16.72  | 0.07 | 16.13     |      |       |
|                 | $G^2 + G_u^2$    | 0.50 | 13.07  | 0.18 | 17.06  | 0.02 | 17.42  | 0.00 | 16.44  | 1.09 | 16.79  | 0.06 | 16.17     |      |       |
| <b>N = 100K</b> | $G^2$            | 7.00 | 5.00   | 5.21 | 7.09   | 3.97 | 6.17   | 4.84 | 7.66   | 4.67 | 8.10   | 2.74 | 7.75      | 0.01 | 8.65  |
|                 | $G_{cc}^2$       | 0.01 | 10.96  | 0.01 | 11.01  | 0.00 | 11.03  | 0.01 | 10.97  | 0.02 | 11.14  | 0.02 | 11.43     |      |       |
|                 | $G_u^2$          | 0.01 | 10.96  | 0.01 | 11.08  | 0.00 | 11.10  | 0.00 | 10.99  | 0.00 | 11.26  | 0.01 | 11.44     |      |       |
|                 | $G^2 + G_{cc}^2$ | 0.00 | 5.00   | 0.16 | 8.89   | 0.15 | 8.78   | 0.00 | 7.66   | 0.44 | 9.09   | 0.49 | 8.76      |      |       |
|                 | $G^2 + G_u^2$    | 0.00 | 5.00   | 0.16 | 9.31   | 0.15 | 8.94   | 0.00 | 7.73   | 0.43 | 9.15   | 0.49 | 8.76      |      |       |

out post-processing identify fewer extra edges at the cost of missing some edges. This happens both, because they are conservative and because of lower power than their i.i.d. counterparts, as we saw in previous experiments. Again, this improves with more samples but they do not seem to significantly outperform the  $G^2$  test, at least in those experiments. Increasing the significance level may improve the situation.

**PC with  $G^2$  test + post-processing with  $G_{cc}^2$  and  $G_u^2$  tests.** The best results, in terms of total number of errors, are achieved when post-processing is applied. For  $N = 1000$  they perform similarly to the  $G^2$  method without post-processing. This is expected, as the first step does not identify many extra edges and thus, post-processing is rarely applied. For larger sample sizes almost no spurious edges are identified, but a few more edges than  $G^2$  without post-processing are missed. This happens because the post-processing rule is erroneously applied to edges that are not due to spurious dependencies and removes them.

Compared to the previous two methods without post-processing, slightly more edges are found but fewer edges are missed. This agrees with the simulations on the simple collider and chain graphs, which showed that the  $G^2$  test is more powerful and therefore misses fewer edges than the  $G_u^2$  and  $G_{cc}^2$  tests.

Finally, the results are similar to the reference results, demonstrating the effectiveness of the proposed methods.

**Comparison of  $G_u^2$  and  $G_{cc}^2$ .** In all simulations the  $G_u^2$  test performs very similar to the  $G_{cc}^2$  test, with the latter being marginally better on average. Note however that averaging may hide the variance of the  $G_u^2$  test. In any case, undersampling is an alternative that can be generalized to other types of data, and should be further investigated.

## 8 CONCLUSION

We proposed methods to learn Bayesian networks from discrete, unmatched case-control data. We showed that one can first learn a network by ignoring the case-control sampling and then apply a post-processing step to remove spurious edges using a specialized test for case-control data. To do this the joint distribution of the selection variables must be available. In case it is not correctly specified the tests may fail to remove spurious edges. Finally, the trivial approach of undersampling seems to be a reasonable alternative, with the advantage that it easily generalizes to other types of data, such as continuous data with discrete selection variables. A drawback however is that it exhibits a large variance as it highly depends on the selected samples.

There is a lot of room for improvement and extensions. First, the proposed post-processing method could be improved to reduce the number of false removals of edges. Second, it is important to investigate additional case-control samplings, such as those from matched or nested studies. Finally, devising methods for other types of data, such as continuous data, would further broaden the scope. One possible approach would be to use or extend the ideas by Kuroki and Cai [2006] for recovering the population covariance matrix. Another possibility is to find a way to perform undersampling multiple times and combine the results appropriately.

### Acknowledgements

We would like to thank Greg Cooper, Sofia Triantafyllou and the anonymous reviewers for their comments. This work was partially funded by the ERC Consolidator Grant No 617393 CAUSALPATH and the Greek GSRT ARIS-TEIA II No 3446 Epilogeas.

## References

- E. Bareinboim, J. Tian, and J. Pearl. Recovering from Selection Bias in Causal and Statistical Inference. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI-14)*, 2014.
- T. Barrett, S. Wilhite, P. Ledoux, C. Evangelista, I. Kim, M. Tomashevsky, K. Marshall, K. Phillippy, P. Sherman, M. Holko, A. Yefanov, H. Lee, N. Zhang, C. Robertson, N. Serova, S. Davis, and A. Soboleva. NCBI GEO: archive for functional genomics data sets—update. *Nucleic Acids Res.*, 41(Database issue):D991–5, January 2013.
- J. Binder, D. Koller, S. Russell, and K. Kanazawa. Adaptive probabilistic networks with hidden variables. *Machine Learning*, 29(2-3):213–244, Nov. 1997.
- N. Breslow. Statistics in epidemiology: The case-control study. *Journal of the American Statistical Association*, 91:14–28, March 1996.
- N. Breslow and N. Day. *Statistical Methods in Cancer Research. Vol. 1 The Analysis of Case-Control Studies*. IARC, Lyon, 1980.
- G. F. Cooper. A Bayesian Method for Causal Modeling and Discovery Under Selection. In *Proceedings of the 16th International Conference on Uncertainty in Artificial Intelligence (UAI 2000)*, 2000.
- G. F. Cooper and C. Yoo. Causal discovery from a mixture of experimental and observational data. In *Proceedings of the 15th International Conference on Uncertainty in Artificial Intelligence (UAI 1999)*, 1999.
- R. Edgar, M. Domrachev, and A. E. Lash. Gene Expression Omnibus: NCBI gene expression and hybridization array data repository. *Nucleic Acids Res.*, 30(1):207–10, January 2002.
- P. Good. *Permutation, Parametric, and Bootstrap Tests of Hypotheses*. Springer Series in Statistics. Springer, 3rd edition, 2004.
- D. Heckerman and J. S. Breese. A new look at causal independence. In *Proceedings of the 10th International Conference on Uncertainty in Artificial Intelligence (UAI 1994)*, 1994.
- A. Hyttinen, P. O. Hoyer, F. Eberhardt, and M. J. R. J. Discovering cyclic causal models with latent variables: A general sat-based procedure. In *Proceedings of the 29th International Conference on Uncertainty in Artificial Intelligence (UAI 2013)*, 2013.
- M. Kuroki and Z. Cai. On recovering a population covariance matrix in the presence of selection bias. *Biometrika*, 93(3):601–611, 2006.
- K. Murphy. *The Bayes Net Toolbox for MATLAB*. *Computing science and statistics*, 2001.
- J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.
- J. Pearl. *Causality, Models, Reasoning, and Inference*. Cambridge University Press, New York, NY, USA, 2000.
- T. Richardson and P. Spirtes. Ancestral graph Markov models. *Annals of Statistics*, 30(4):962–1030, 2002.
- K. J. Rothman, S. Greenland, and T. L. Lash. *Modern Epidemiology*. Williams & Wilkins, Philadelphia, PA: Lippincott, 3rd edition, 2008.
- P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. MIT Press, Cambridge, MA, 2nd edition, 2000.
- S. Srinivas. A generalization of the noisy-or model. In *Proceedings of the Ninth International Conference on Uncertainty in Artificial Intelligence (UAI 1993)*, 1993.
- R. E. Tillman and P. Spirtes. Learning equivalence classes of acyclic models with latent and selection variables from multiple datasets with overlapping variables. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS 2011)*, 2011.
- S. Triantafillou and I. Tsamardinos. Constraint-based causal discovery from multiple interventions over overlapping variable sets. *CoRR*, abs/1403.2150, 2014.
- I. Tsamardinos and G. Borboudakis. Permutation Testing Improves Bayesian Network Learning. In *Proceedings of the 2010 European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD 2010)*, 2010.
- I. Tsamardinos, L. Brown, and C. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65(1):31–78, 2006.
- A. Zagorecki and M. Druzdzal. Knowledge engineering for Bayesian networks: How common are noisy-max distributions in practice? In *Proceedings of 17th European Conference on Artificial Intelligence (ECAI 2006)*, 2006.

---

# Efficient Algorithms for Bayesian Network Parameter Learning from Incomplete Data

---

Guy Van den Broeck\* and Karthika Mohan\* and Arthur Choi and Adnan Darwiche and Judea Pearl

Computer Science Department

University of California, Los Angeles

{guyvdb, karthika, aychoi, darwiche, judea}@cs.ucla.edu

## Abstract

We propose a family of efficient algorithms for learning the parameters of a Bayesian network from incomplete data. Our approach is based on recent theoretical analyses of missing data problems, which utilize a graphical representation, called the missingness graph. In the case of MCAR and MAR data, this graph need not be explicit, and yet we can still obtain closed-form, asymptotically consistent parameter estimates, without the need for inference. When this missingness graph is explicated (based on background knowledge), even partially, we can obtain even more accurate estimates with less data. Empirically, we illustrate how we can learn the parameters of large networks from large datasets, which are beyond the scope of algorithms like EM (which require inference).

## 1 INTRODUCTION

When learning the parameters of a Bayesian network from data with missing values, the conventional wisdom among machine learning practitioners is that there are two options: use *expectation maximization* (EM) or *gradient methods* (to optimize the likelihood); see, e.g., Darwiche (2009), Koller and Friedman (2009), Murphy (2012), Barber (2012). Both of these approaches, however, suffer from the following disadvantages, which prevent them from scaling to large networks and datasets; see also Thiesson, Meek, and Heckerman (2001). First, they are *iterative*, and hence may need many passes over a potentially large dataset. Next, these algorithms may get stuck in *local optima*, which means that, in practice, one must run these algorithms multiple times with different initial seeds, and hope that one of them leads to a good optimum. Last, but not least, these methods *require inference* in the network, which places a hard limit

---

\*Both authors contributed equally to this work. Gvdb is also affiliated with KU Leuven, Belgium.

on the networks where EM and gradient methods can even be applied, namely for networks where exact inference is tractable, i.e., they have small enough treewidth, or sufficient local structure (Chavira & Darwiche, 2006, 2007).

Recently, Mohan, Pearl, and Tian (2013) showed that the joint distribution of a Bayesian network is recoverable from incomplete data, including data that falls under the classical *missing at random* assumption (MAR), but also for a broad class of data that is not MAR. Their analysis is based on a graphical representation for missing data problems, called the *missingness graph*, where one explicates the *causal mechanisms* that are responsible for the missingness in an incomplete dataset. Using this representation, they provide a way to decide whether a given query (e.g., a joint marginal) is recoverable, and if so, they provide a *closed-form* expression (in terms of the observables) for an asymptotically consistent estimate.

Building on the theoretical foundations set by Mohan et al. (2013), we propose a family of practical and efficient algorithms for estimating the parameters of a Bayesian network from incomplete data. For the cases of both MCAR and MAR data, where the missingness graph need not be explicit, we start by deriving the closed-form parameter estimates, as implied by Mohan et al. (2013). We next show how to obtain better estimates, by exploiting a factorized representation that allows us to aggregate distinct, yet asymptotically equivalent estimates, hence utilizing more of the data. We also show how to obtain improved estimates, when the missingness graph is only partially explicated (based on domain or expert knowledge). As in Mohan et al. (2013), all of our estimation algorithms are asymptotically consistent, i.e., they converge to the true parameters of a network, in the limit of infinite data.

As we show empirically, our parameter estimation algorithms make learning from incomplete data viable for larger Bayesian networks and larger datasets, that would otherwise be beyond the scope of algorithms such as EM and gradient methods. In particular, our algorithms (1) are non-iterative, requiring only a single pass over the data, (2) provide estimates in closed-form, and hence do not suffer from

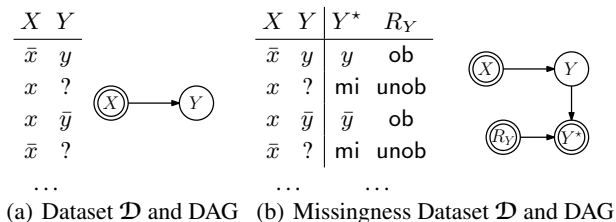


Figure 1: Datasets and DAGs.

local optima, and (3) require no inference, which is the primary limiting factor for the scalability of algorithms such as EM. We note that these advantages are also available when learning Bayesian networks from *complete* data.

## 2 TECHNICAL PRELIMINARIES

In this paper, we use upper case letters ( $X$ ) to denote variables and lower case letters ( $x$ ) to denote their values. Variable sets are denoted by bold-face upper case letters ( $\mathbf{X}$ ) and their instantiations by bold-face lower case letters ( $\mathbf{x}$ ). Generally, we will use  $X$  to denote a variable in a Bayesian network and  $\mathbf{U}$  to denote its parents. A network parameter will therefore have the general form  $\theta_{x|\mathbf{u}}$ , representing the probability  $\Pr(X=x|\mathbf{U}=\mathbf{u})$ .

Given an incomplete dataset  $\mathcal{D}$ , we want to learn the parameters of the Bayesian network  $\mathcal{N}$  that the dataset originated from. This network induces a distribution  $\Pr(\mathbf{X})$ , which is in general unknown; instead, we only have access to the dataset  $\mathcal{D}$ .

### 2.1 MISSING DATA: AN EXAMPLE

As an illustrative example, consider Figure 1(a), depicting a dataset  $\mathcal{D}$ , and the directed acyclic graph (DAG)  $\mathcal{G}$  of a Bayesian network, both over variables  $X$  and  $Y$ . Here, the value for variable  $X$  is always observed in the data, while the value for variable  $Y$  can be missing. In the graph, we denote a variable that is always observed with a double-circle. Now, if we happen to know the mechanism that causes the value of  $Y$  to become missing in the data, we can include it in our model, as in Figure 1(b). Here, we use a variable  $R_Y$  to represent the mechanism that controls whether the value of variable  $Y$  is missing or observed. Further, we witness the value of  $Y$ , or its missingness, through a proxy variable  $Y^*$ , as an observation. Such a graph, which explicates the missing data process, is called a *missingness graph*.

In our example, we augmented the dataset and graph with new variables  $R_Y$ , representing the causal mechanism that dictates the missingness of the value of  $Y$ . This mechanism can be active ( $Y$  is unobserved), denoted by  $R_Y=\text{unob}$ . Otherwise, the mechanism is passive ( $Y$  is observed), de-

noted by  $R_Y=\text{ob}$ . Variable  $Y^*$  acts as a proxy on the value of  $Y$ , which may be an observed value  $y$ , or a special value (mi) when the value of  $Y$  is missing. The value of  $Y^*$  thus depends functionally on variables  $R_Y$  and  $Y$ :

$$Y^* = f(R_Y, Y) = \begin{cases} \text{mi} & \text{if } R_Y = \text{unob} \\ Y & \text{if } R_Y = \text{ob} \end{cases}$$

That is, when  $R_Y=\text{unob}$ , then  $Y^*=\text{mi}$ ; otherwise  $R_Y=\text{ob}$  and the proxy  $Y^*$  assumes the observed value of  $Y$ .

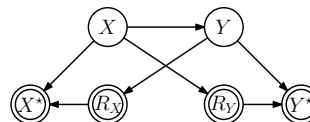


Figure 2: An MNAR missingness graph.

Figure 2 highlights a more complex example of a missingness graph, with causal mechanisms  $R_X$  and  $R_Y$  that depend on other variables. In Section 2.3, we highlight how different missing data problems (such as MCAR and MAR) lead to different types of missingness graphs.

### 2.2 LEARNING WITH MISSINGNESS GRAPHS

Given a Bayesian network with DAG  $G$ , and an incomplete dataset  $\mathcal{D}$ , we can partition the variables  $\mathbf{X}$  into two sets: the fully-observed variables  $\mathbf{X}_o$ , and the partially-observed variables  $\mathbf{X}_m$  that have missing values in  $\mathcal{D}$ . As in our example above, one can take into account knowledge about the processes responsible for the missingness in  $\mathcal{D}$ . More specifically, we can incorporate the causal mechanisms that cause the variables  $\mathbf{X}_m$  to have missing values, by introducing (1) variables  $\mathbf{R}$  representing the *causal mechanisms* that are responsible for missingness in the data, and (2) variables  $\mathbf{X}_m^*$  that act as *proxies* to the variables  $\mathbf{X}_m$ . This augmented Bayesian network, which we refer to as the *missingness graph*  $\mathcal{N}^*$ , has variables  $\mathbf{X}_o, \mathbf{X}_m^*, \mathbf{R}$  that are fully-observed, and variables  $\mathbf{X}_m$  that are only partially-observed. The missingness graph  $\mathcal{N}^*$  thus induces a distribution  $\Pr(\mathbf{X}_o, \mathbf{X}_m, \mathbf{X}_m^*, \mathbf{R})$ . Using the missingness graph, we want to draw conclusions about the partially-observed variables, by reasoning about the fully-observed ones.

Missingness graphs can serve as a powerful tool for analyzing missing data problems; see, e.g., Thoemmes and Mohan (2015), Francois and Leray (2007), Darwiche (2009), Koller and Friedman (2009). As Mohan et al. (2013) show, one can exploit the conditional independencies that missingness graphs encode, in order to extract *asymptotically consistent* estimates for missing data problems, including MNAR ones, whose underlying assumptions would put it out of the scope of existing techniques.<sup>1</sup> Mohan et al.

<sup>1</sup>Note that maximum-likelihood estimation is asymptotically consistent, although a consistent estimator is not necessarily a maximum-likelihood estimator; see, e.g., Wasserman (2011).

(2013) identify conditions on  $\mathcal{N}^*$  that allow the original, partially-observed distribution  $\Pr(\mathbf{X}_o, \mathbf{X}_m)$  to be identified from the fully-observed distribution  $\Pr(\mathbf{X}_o, \mathbf{X}_m^*, \mathbf{R})$ . However, in practice, we only have access to a dataset  $\mathcal{D}$ , and the corresponding *data distribution* that it induces:

$$\Pr_{\mathcal{D}}(\mathbf{x}_o, \mathbf{x}_m^*, \mathbf{r}) = \frac{1}{N} \mathcal{D}\#(\mathbf{x}_o, \mathbf{x}_m^*, \mathbf{r}),$$

where  $N$  is the number of instances in dataset  $\mathcal{D}$ , and where  $\mathcal{D}\#(\mathbf{x})$  is the number of instances where instantiation  $\mathbf{x}$  appears in the data.<sup>2</sup> However, the data distribution  $\Pr_{\mathcal{D}}$  tends to the true distribution  $\Pr$  (over the fully-observed variables), as  $N$  tends to infinity.

Building on the theoretical foundations set by Mohan et al. (2013), we shall propose a family of efficient and scalable parameter estimation algorithms from incomplete data. In essence, we will show how to query the observed data distribution  $\Pr_{\mathcal{D}}$ , in order to make inferences about the true, underlying distribution  $\Pr(\mathbf{X}_o, \mathbf{X}_m)$  (in particular, we want the conditional probabilities that parameterize the given Bayesian network). As we shall discuss, in many cases the missingness graph need not be explicit. In other cases, when there is knowledge about the missingness graph, even just partial knowledge, we can exploit it, in order to obtain more accurate parameter estimates.

### 2.3 CATEGORIES OF MISSINGNESS

An incomplete dataset is categorized as *Missing Completely At Random* (MCAR) if all mechanisms  $\mathbf{R}$  that cause the values of variables  $\mathbf{X}_m$  to go missing, are marginally independent of  $\mathbf{X}$ , i.e., where  $(\mathbf{X}_m, \mathbf{X}_o) \perp\!\!\!\perp \mathbf{R}$ . This corresponds to a missingness graph where no variable in  $\mathbf{X}_m \cup \mathbf{X}_o$  is a parent of any variable in  $\mathbf{R}$ . For example, if all mechanisms  $\mathbf{R}$  are root nodes, then the problem is MCAR. Note that the missingness graph of Figure 1(b) implies an MCAR dataset.

An incomplete dataset is categorized as *Missing At Random* (MAR) if missingness mechanisms are conditionally independent of the partially-observed variables given the fully-observed variables, i.e., if  $\mathbf{X}_m \perp\!\!\!\perp \mathbf{R} \mid \mathbf{X}_o$ . This corresponds to a missingness graph where variables  $\mathbf{R}$  are allowed to have parents, as long as none of them are partially-observed. In the example missingness graph of Figure 1(b), adding an edge  $X \rightarrow R_Y$  results in a graph that yields MAR data. This is a stronger, variable-level definition of MAR, which has previously been used in the machine learning literature (Darwiche, 2009; Koller & Friedman, 2009), in contrast to the event-level definition of MAR that is prevalent in the statistics literature (Rubin, 1976).

<sup>2</sup>Note that the data distribution is well-defined over the variables  $\mathbf{X}_o, \mathbf{X}_m^*$ , and  $\mathbf{R}$ , as they are fully-observed in the augmented dataset, and that  $\Pr_{\mathcal{D}}$  can be represented compactly in space linear in  $N$ , as we need not explicitly represent those instantiations  $\mathbf{x}$  that were not observed in the data.

Table 1: Summary of Estimation Algorithms

| Algorithm | Description (Section Number)                  |
|-----------|---|
| D-MCAR    | Direct Deletion for MCAR data (3.1)           |
| D-MAR     | Direct Deletion for MAR data (3.2)            |
| F-MCAR    | Factored Deletion for MCAR data (3.3)         |
| F-MAR     | Factored Deletion for MAR data (3.3)          |
| I-MAR     | Informed Deletion for MAR data (5.1)          |
| IF-MAR    | Informed Factored Deletion for MAR data (5.1) |

An incomplete dataset is categorized as *Missing Not At Random* (MNAR) if it is not MAR (and thus not MCAR). For example, the DAG in Figure 2 corresponds to an MNAR missingness graph. This is because the mechanism  $R_X$  has a partially-observed variable as a parent; further, mechanism  $R_Y$  has a partially-observed parent  $X$ .

## 3 CLOSED-FORM LEARNING

We now present algorithms to learn the parameters of a Bayesian network  $\mathcal{N}$  from data  $\mathcal{D}$ . We first consider the classical missing data assumptions, with no further knowledge about the missingness graph that generated the data.

To estimate the conditional probabilities  $\theta_{x|\mathbf{u}}$  that parameterize a Bayesian network, we estimate the joint distributions  $\Pr(X, \mathbf{U})$ , which are subsequently normalized, as a conditional probability table. Hence, it suffices, for our discussion, to estimate marginal distributions  $\Pr(\mathbf{Y})$  for families  $\mathbf{Y} = \{X\} \cup \mathbf{U}$ . We let  $\mathbf{Y}_o = \mathbf{Y} \cap \mathbf{X}_o$  denote the observed variables in  $\mathbf{Y}$ , and  $\mathbf{Y}_m = \mathbf{Y} \cap \mathbf{X}_m$  denote the partially-observed variables. Further, we let  $\mathbf{R}_{\mathbf{Z}} \subseteq \mathbf{R}$  denote the missingness mechanisms for the partially-observed variables  $\mathbf{Z}$ . Through  $\mathcal{D}$ , we have access to the data distribution  $\Pr_{\mathcal{D}}$  over the variables in the missingness dataset. Appendix D illustrates our learning algorithms on a concrete dataset and Table 1 gives an overview of the different estimation algorithms in this paper.

### 3.1 DIRECT DELETION FOR MCAR

The statistical technique of *listwise deletion* is perhaps the simplest technique for performing estimation with MCAR data: we simply delete all instances in the dataset that contain missing values, and estimate our parameters from the remaining dataset, which is now complete. Of course, with this technique, we potentially ignore large parts of the dataset. The next simplest technique is perhaps pairwise deletion, or available-case analysis: when estimating a quantity over a pair of variables  $X$  and  $Y$ , we delete just those instances where variable  $X$  or variable  $Y$  is missing.

Consider now the following, more general, deletion technique, which is expressed in the terms of causal missingness mechanisms. In particular, to estimate the marginals  $\Pr(\mathbf{Y})$  of a set of (family) variables  $\mathbf{Y}$ , from the data dis-

tribution  $\Pr_{\mathcal{D}}$ , we can use the estimate:

$$\begin{aligned} \Pr(\mathbf{Y}) &= \Pr(\mathbf{Y}_o, \mathbf{Y}_m | \mathbf{R}_{\mathbf{Y}_m} = \text{ob}) \quad \text{by } \mathbf{X}_o, \mathbf{X}_m \perp\!\!\!\perp \mathbf{R} \\ &= \Pr(\mathbf{Y}_o, \mathbf{Y}_m^* | \mathbf{R}_{\mathbf{Y}_m} = \text{ob}) \quad \text{by } \mathbf{X}_m = \mathbf{X}_m^* \text{ when } \mathbf{R} = \text{ob} \\ &\approx \Pr_{\mathcal{D}}(\mathbf{Y}_o, \mathbf{Y}_m^* | \mathbf{R}_{\mathbf{Y}_m} = \text{ob}) \end{aligned}$$

That is, we can estimate  $\Pr(\mathbf{Y})$  by using the subset of the data where every variable in  $\mathbf{Y}$  is observed (which follows from the assumptions implied by MCAR data). Since the data distribution  $\Pr_{\mathcal{D}}$  tends to the true distribution  $\Pr$ , this implies a consistent estimate for the marginals  $\Pr(\mathbf{Y})$ . In contrast, the technique of listwise deletion corresponds to the estimate  $\Pr(\mathbf{Y}) \approx \Pr_{\mathcal{D}}(\mathbf{Y}_o, \mathbf{Y}_m^* | \mathbf{R}_{\mathbf{X}_m} = \text{ob})$ , and the technique of pairwise deletion corresponds to the above, when  $\mathbf{Y}$  contains two variables. To facilitate comparisons with more interesting estimation algorithms that we shall subsequently consider, we refer to the more general estimation approach above as *direct deletion*.

### 3.2 DIRECT DELETION FOR MAR

In the case of MAR data, we cannot use the simple deletion techniques that we just described for MCAR data—the resulting estimates would not be consistent. However, we show next that it is possible to obtain consistent estimates from MAR data, using a technique that is as simple and efficient as direct deletion. Roughly, we can view this technique as deleting certain instances from the dataset, but then re-weighting the remaining ones, so that a consistent estimate is obtained. We shall subsequently show how to obtain even better estimates by factorization.

Again, to estimate network parameters  $\theta_{x|u}$ , it suffices to show how to estimate family marginals  $\Pr(\mathbf{Y})$ , now under the MAR assumption. Let  $\mathbf{X}'_o = \mathbf{X}_o \setminus \mathbf{Y}_o$  denote the fully-observed variables outside of the family variables  $\mathbf{Y}$  (i.e.,  $\mathbf{X}_o = \mathbf{Y}_o \cup \mathbf{X}'_o$ ). We have

$$\begin{aligned} \Pr(\mathbf{Y}) &= \sum_{\mathbf{X}'_o} \Pr(\mathbf{Y}_o, \mathbf{Y}_m, \mathbf{X}'_o) \\ &= \sum_{\mathbf{X}'_o} \Pr(\mathbf{Y}_m | \mathbf{Y}_o, \mathbf{X}'_o) \Pr(\mathbf{Y}_o, \mathbf{X}'_o) \end{aligned}$$

Hence, we reduced the problem to estimating two sets of probabilities. Estimating the probabilities  $\Pr(\mathbf{Y}_o, \mathbf{X}'_o)$  is straightforward, as variables  $\mathbf{Y}_o$  and  $\mathbf{X}'_o$  are fully observed in the data. The conditional probabilities  $\Pr(\mathbf{Y}_m | \mathbf{Y}_o, \mathbf{X}'_o)$  contain partially observed variables  $\mathbf{Y}_m$ , but they are conditioned on all fully observed variables  $\mathbf{X}_o = \mathbf{Y}_o \cup \mathbf{X}'_o$ . The MAR definition implies that each subset of the data that fixes a value for  $\mathbf{X}_o$  is locally MCAR. Like the MCAR case, we can estimate each conditional probability as

$$\Pr(\mathbf{Y}_m | \mathbf{Y}_o, \mathbf{X}'_o) = \Pr(\mathbf{Y}_m^* | \mathbf{Y}_o, \mathbf{X}'_o, \mathbf{R}_{\mathbf{Y}_m} = \text{ob}).$$

This leads to the following estimation algorithm,

$$\Pr(\mathbf{Y}) \approx \sum_{\mathbf{X}'_o} \Pr_{\mathcal{D}}(\mathbf{Y}_m^* | \mathbf{Y}_o, \mathbf{X}'_o, \mathbf{R}_{\mathbf{Y}_m} = \text{ob}) \Pr_{\mathcal{D}}(\mathbf{Y}_o, \mathbf{X}'_o)$$

---

#### Algorithm 1 F-MCAR( $y, \mathcal{D}$ )

---

**Input:**

$y$ : A state of query variables  $\mathbf{Y}$

$\mathcal{D}$ : An incomplete dataset with data distribution  $\Pr_{\mathcal{D}}$

**Auxiliary:**

CACHE: A global cache of estimated probabilities

**Function:**

- 1: **if**  $y = \emptyset$  **then return** 1
  - 2: **if**  $\text{CACHE}[y] \neq \text{nil}$  **then return**  $\text{CACHE}[y]$
  - 3:  $\mathcal{E} \leftarrow \emptyset$  // Initialize set of estimates
  - 4: **for each**  $y \in \mathbf{y}$  **do**
  - 5:    $\mathbf{u} \leftarrow \mathbf{y} \setminus \{y\}$  // Factorize with parents  $\mathbf{u}$
  - 6:   **add**  $\Pr_{\mathcal{D}}(y | \mathbf{u}, \mathbf{R}_y = \text{ob}) \cdot \text{F-MCAR}(\mathbf{u}, \mathcal{D})$  **to**  $\mathcal{E}$
  - 7:  $\text{CACHE}[y] \leftarrow \text{Aggregate estimates in } \mathcal{E}$  // E.g., mean
  - 8: **return**  $\text{CACHE}[y]$
- 

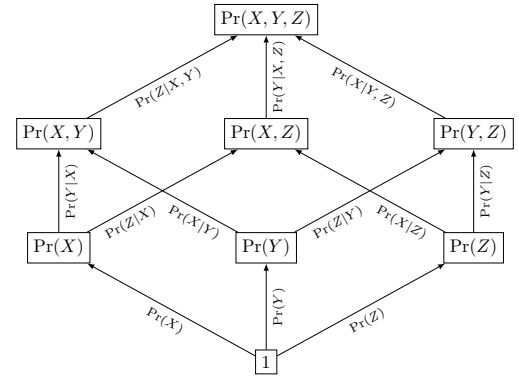


Figure 3: Factorization Lattice of  $\Pr(X, Y, Z)$

which uses only the fully-observed variables of the data distribution  $\Pr_{\mathcal{D}}$ . Note that the summation requires only a single pass through the data, i.e., for only those instantiations of  $\mathbf{X}'_o$  that appear in it. Again,  $\Pr_{\mathcal{D}}$  tends to the true distribution  $\Pr$ , as the dataset size tends to infinity, implying a consistent estimate of  $\Pr(\mathbf{Y})$ .

### 3.3 FACTORED DELETION

We now propose a class of deletion algorithms that exploit more data than direct deletion. In the first step, we generate multiple but consistent estimates for the query so that each estimate utilizes different parts of a dataset to estimate the query. In the second step, we aggregate these estimates to compute the final estimate and thus put to use almost all tuples in the dataset. Since this method exploits more data than direct deletion, it obtains a better estimate of the query.

**Factored Deletion for MCAR** Algorithm 1 implements factored deletion for MCAR. Let the query of interest be  $\Pr(\mathbf{Y})$ , and let  $Y^1, Y^2, \dots, Y^n$  be any ordering of the  $n$

variables in  $\mathbf{Y}$ . Each ordering yields a unique factorization:

$$\Pr(\mathbf{Y}) = \prod_{i=1}^n \Pr(Y^i | Y^{i+1}, \dots, Y^n)$$

We can estimate each of these factors independently, on the subset of the data in which all of its variables are fully observed (as in direct deletion), i.e.,

$$\Pr(Y^i | Y^{i+1}, \dots, Y^n) = \Pr(Y^i | Y^{i+1}, \dots, Y_m^n, \mathbf{R}_{\mathbf{Z}^i} = \text{ob})$$

where  $\mathbf{Z}^i$  is the set of partially-observed variables in the factor. When  $|\mathbf{Y}_m| > 1$ , we can utilize much more data than direct deletion. See Appendix D, for an example.

So far, we have discussed how a consistent estimate of  $\Pr(\mathbf{Y})$  may be computed given a factorization. Now we shall detail how estimates from each factorization can be aggregated to compute more accurate estimates of  $\Pr(\mathbf{Y})$ . Let  $k$  be the number of variables in a family  $\mathbf{Y}$ . The number of possible factorizations is  $k!$ . However, different factorizations share the same sub-factors, which we can estimate once, and reuse across factorizations. We can organize these computations using a lattice, as in Figure 3, which has only  $2^k$  nodes and  $k \cdot 2^{k-1}$  edges. Our algorithm will compute as many estimates as there are edges in this lattice, which is only on the order of  $O(n \log n)$ , where  $n$  is the number of parameters being estimated for a family  $Y$  (which is also exponential in the number of variables  $k$ ). To emphasize the distinction with direct deletion, which uses only those instances in the data where *all* variables in  $\mathbf{Y}$  are observed, factored deletion uses any instance in the data where *at least one* variable in  $\mathbf{Y}$  is observed.

More specifically, our factored deletion algorithm first estimates the conditional probabilities on the edges of the lattice, each estimate using the subset of the data where its variables are observed. Second, it propagate the estimates, bottom-up. For each node, there are several alternative estimates available, on its incoming edges. There are various ways of aggregating these estimates, such as mean, median, and propagating the lowest-variance estimate.<sup>3</sup>

**Factored Deletion for MAR** Algorithm 2 implements factored deletion for MAR. Let  $Y_m^1, Y_m^2, \dots, Y_m^n$  be any ordering of the  $n$  partially observed variables  $\mathbf{Y}_m \subseteq \mathbf{Y}$  and let  $\mathbf{X}'_o = \mathbf{X}_o \setminus \mathbf{Y}_o$  denote the fully-observed variables outside of  $\mathbf{Y}$ . Given an ordering, we have the factorization:

$$\Pr(\mathbf{Y}) = \sum_{\mathbf{X}'_o} \Pr(\mathbf{Y}_o, \mathbf{X}'_o) \prod_{i=1}^n \Pr(Y_m^i | \mathbf{Z}^{i+1}, \mathbf{X}_o)$$

where  $\mathbf{Z}^i_m = \{Y_m^j | i \leq j \leq n\}$ . We then proceed in a manner similar to factored deletion for MCAR to estimate individual factors and aggregate estimates to compute  $\Pr(\mathbf{Y})$ . For equations and derivations, please see Appendix A.

<sup>3</sup>In initial experiments, all aggregations performed similarly. Reported results use an inverse-variance weighting heuristic.

---

## Algorithm 2 F-MAR( $\mathbf{y}, \mathcal{D}$ )

---

**Input:**

$\mathbf{y}$ : A state of query variables  $\mathbf{Y}$ , consisting of  $\mathbf{y}_o$  and  $\mathbf{y}_m$   
 $\mathcal{D}$ : An incomplete dataset with data distribution  $\Pr_{\mathcal{D}}$

**Function:**

```

1:  $e \leftarrow 0$  // Estimated probability
2: for each  $\mathbf{x}_o$  appearing in  $\mathcal{D}$  that agrees with  $\mathbf{y}_o$  do
3:    $\mathcal{D}_{\mathbf{x}_o} \leftarrow$  subset of  $\mathcal{D}$  where  $\mathbf{x}_o$  holds
4:    $e \leftarrow e + \Pr_{\mathcal{D}}(\mathbf{x}_o) \cdot \text{F-MCAR}(\mathbf{y}_m, \mathcal{D}_{\mathbf{x}_o})$ 
5: return  $e$ 

```

---

## 4 EMPIRICAL EVALUATION

To evaluate the learning algorithms we proposed, we simulate partially observed datasets from Bayesian networks, and re-learn their parameters from the data.<sup>4</sup>

In our first sets of experiments, we compare our parameter estimation algorithms with EM, on relatively small networks for MCAR and MAR data. These experiments are intended to observe general trends in our algorithms, in terms of their computational efficiency, but also in terms of the quality of the parameter estimates obtained. Our main empirical contributions are presented in Section 4.3, where we demonstrate the scalability of our proposed estimation algorithms, to larger networks and datasets, compared to EM (even when using approximate inference algorithms).

We consider the following algorithms:

**D-MCAR & F-MCAR:** direct deletion and factored deletion for MCAR data.

**D-MAR & F-MAR:** direct deletion and factored deletion for MAR data.

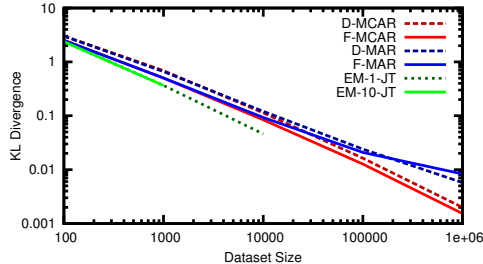
**EM- $k$ -JT:** EM with  $k$  random restarts, jointree inference.

**F-MAR + EM-JT:** EM seeded with F-MAR estimates, jointree inference.

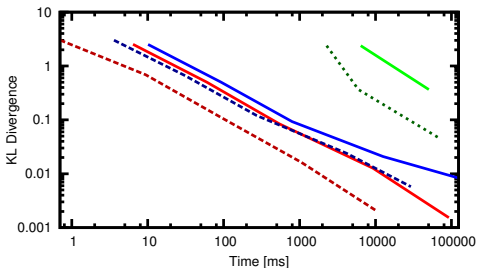
Remember that D-MCAR and F-MCAR are consistent for MCAR data only, while D-MAR and F-MAR are consistent for general MAR data. EM is consistent for MAR data, but only if it converges to maximum-likelihood estimates.

We evaluate the learned parameters in terms of their *likelihood* on independently generated, fully-observed test data, and the *Kullback–Leibler divergence* (KLD) between the original and learned Bayesian networks. We report per-instance log-likelihoods (which are divided by dataset size). We evaluate the learned models on unseen data, so all learning algorithms assume a symmetric Dirichlet prior

<sup>4</sup>An implementation of our system is available at <http://reasoning.cs.ucla.edu/deletion>.



(a) KL Divergence vs. Dataset Size



(b) KL Divergence vs. Time

Figure 4: Learning the `alarm` network from MCAR data.

on the network parameters with a concentration parameter of 2 (which corresponds to Laplace smoothing).

#### 4.1 MCAR DATA

First, we consider learning from *MCAR data*, evaluating the quality of the parameters learned by each algorithm. We simulate training sets of increasing size, from a given Bayesian network, selecting 30% of the variables to be partially observed, and removing 70% of their values completely at random. All reported numbers are averaged over 32 repetitions with different learning problems. When no number is reported, a 5 minute time limit was exceeded.

To illustrate the trade-off between data and computational resources, Figure 4 plots the KLDs as a function of dataset size and time; further results are provided in Table 5 of Appendix B. First, we note that in terms of the final estimates obtained, there is no advantage in running EM with restarts: EM-1-JT and EM-10-JT learn almost identical models. This indicates that the likelihood landscape for MCAR data has few local optima, and is easy to optimize. Hence, EM may be obtaining maximum-likelihood estimates in these cases. In general, maximum-likelihood estimators are more statistically efficient (asymptotically) than other estimators, i.e., they require fewer samples. However, other estimators (such as method-of-moments) can be more computationally efficient; see, e.g., Wasserman (2011). We also observe this trend here. EM obtains better estimates with smaller datasets, with smaller KLDs. However, direct and factored deletion (D-MCAR and F-MCAR) are both orders-of-magnitude faster, and can scale to much larger

datasets, than EM (which requires inference). Further, F-MCAR needs only a modest amount of additional data to obtain comparable estimates.

To compare our direct and factored methods, we see that F-MCAR is slower than D-MCAR, as it estimates more quantities (one for each lattice edge). F-MCAR learns better models, however, as it uses a larger part of the available data. Finally, D-MAR performs worse than F-MCAR and D-MCAR, as it assumes the weaker MAR assumption. All learners are consistent, as all KLDs converge to zero.

#### 4.2 MAR DATA

Next, we consider the more challenging problem of learning from *MAR data*, which we generate as follows: (1) select an  $m$ -fraction of the variables to be partially observed, (2) add a missingness mechanism variable  $R_X$  for each partially-observed variable  $X$ , (3) assign  $p$  parents to each  $R_X$ , randomly selected from the set of observed variables, giving preference to neighbors of  $X$  in the network, (4) sample parameters for the missingness mechanism CPTs from a Beta distribution, (5) sample a complete dataset with  $R_X$  values, and (6) hide values of  $X$  accordingly.

For our first MAR experiment, we use a small network that is tractable enough for EM to scale to large dataset sizes, so that we can observe trends in this regime. Figure 5(a) shows KLD for the `fire alarm` network, which has only 6 variables (and hence, the complexity of inference is negligible). The missing data mechanisms were generated with  $m = 0.3$ ,  $p = 2$ , and a Beta distribution with shape parameters 1.0 and 0.5. All numbers are averaged over 64 repetitions with different random learning problems.<sup>5</sup>

There is a significant difference between EM, with and without restarts, indicating that the likelihood landscape is challenging to optimize (compared to MCAR, which we just evaluated). EM-10-JT performs well for small dataset sizes, but stops converging after around 1,000 instances. This could be due to all restarts getting stuck in local optima. The KLD of F-MAR starts off between EM-1-JT and EM-10-JT for small sizes, but quickly outperforms EM. For the largest dataset sizes, it learns networks whose KLD is two orders of magnitude smaller than EM-10-JT. The KLD improves further when we use F-MAR estimates to seed EM. This approach is on par with EM-10 for small datasets, while still converging for large dataset sizes. However, note that using F-MAR to seed EM will not be practical for larger networks, where inference becomes a

<sup>5</sup>On our chosen parameters: (1) the number of repetitions was chosen to produce smooth learning curves; (2) a Beta distribution with shape parameter 1 is uniform, and with parameter 0.5, it is slightly biased (so that it acts more like an MAR, and less like an MCAR, mechanism); (3)  $m = 0.3$  corresponds to a low amount of missing data, and later  $m = 0.9$  corresponds to high amount; and (4)  $p = 2$  encourages sparsity and keeps the CPTs small, although setting  $p$  to 1 or 3 does not change the results.



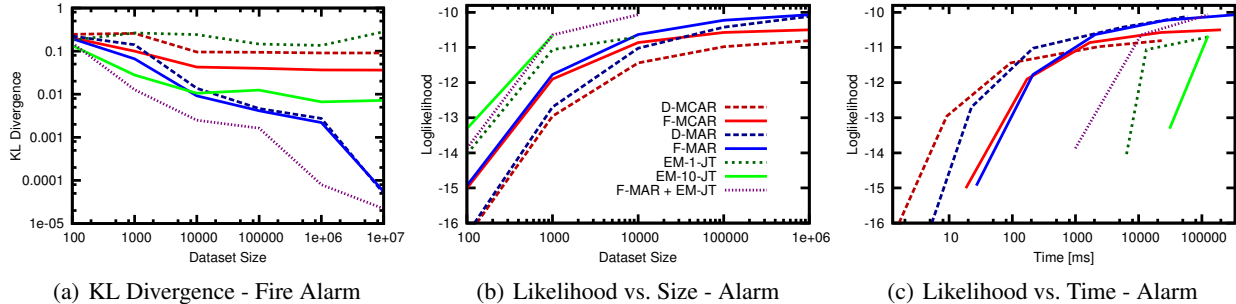


Figure 5: Learning small, tractable Bayesian networks from MAR data. The legend is given in sub-figure (b).

bottleneck. D-MCAR and F-MCAR are not consistent for MAR data, and indeed converge to a biased estimate with a KLD around 0.1. Finally, we observe that the factorized algorithms generally outperform their direct counterparts.

For our second MAR experiment, we work with the classical `alarm` network, which has 37 variables. The missing data mechanisms were generated with  $m = 0.9$ ,  $p = 2$ , and a Beta distribution with shape parameters 0.5. All reported numbers are averaged over 32 repetitions, and when no number is reported, a 10 minute time limit was exceeded.

Figures 5(b) and 5(c) show test set likelihood as a function of dataset *size* and learning *time*. EM-10-JT performs well for very small dataset sizes, and again outperforms EM-1-JT. However, inference time is non-negligible and EM-1-JT fails to scale beyond 1,000 instances, whereas EM-10-JT scales to 10,000 (as one would expect). The closed-form learners dominate all versions of EM as a function of time, and scale to dataset sizes that are two orders of magnitude larger. EM seeded by F-MAR achieves similar quality to EM-10-JT, while being significantly faster than EM learners with random seeds. D-MAR and F-MAR are more computationally efficient, and can scale to much larger dataset sizes. Further, as seen in Figure 5(c), they can obtain good likelihoods even before the EM methods report their first likelihoods.

### 4.3 SCALING TO LARGER NETWORKS

In our last set of experiments of this section, we evaluate our algorithms on their ability to scale to larger networks, with higher treewidths, where exact inference is more challenging.<sup>6</sup> Again, inference is the main factor that limits the scalability of algorithms such as EM, to larger networks and datasets (EM invokes inference as a sub-routine, once per data instance, per iteration). Tables 2 & 3 report results on four networks, where we simulated MAR datasets, as in the previous set of experiments. Each method is given a time limit of 5 or 25 minutes. Appendix C provides results on additional settings. We consider the following methods:

<sup>6</sup>The `grid` network has 400 variables, `munin1` has 189 variables, `water` has 32 variables, and `barley` has 48 variables.

**EM-JT** The EM-10-JT algorithm used in anytime fashion, which returns, given a time limit, the best parameters found in any restart, even if EM did not converge.

**EM-BP** A variant of EM-JT that uses (loopy) belief propagation for (approximate) inference (in the E-step).

We see that EM-JT, which performs exact inference, does not scale well to these networks. This problem is mitigated by EM-BP, which performs *approximate* inference, yet we find that it also has difficulties scaling (dashed entries indicate that EM-JT and EM-BP did not finish 1 iteration of EM). In contrast, F-MAR, and particularly D-MAR, can scale to much larger datasets. This efficiency is due to the relative simplicity of the D-MAR and F-MAR estimation algorithms: they are not iterative and require only a single pass over the data. In contrast, with EM-BP, the EM algorithm is not only iterative, but the BP algorithm that EM-BP invokes as a sub-routine, is itself an iterative algorithm. As for accuracy, F-MAR typically obtains the best likelihoods (in bold) for larger datasets, while EM-BP can perform better on smaller datasets. We also evaluated D-MCAR and F-MCAR, although they are not in general consistent for MAR data. We find that they scale even further, and can also produce good estimates in terms of likelihood.

## 5 EXPLOITING MISSINGNESS GRAPHS

We have so far made very general assumptions about the structure of the missingness graph, capturing the MCAR and MAR assumptions. In this section, we show how to exploit additional knowledge about the missingness graph to further improve the quality of our estimates. Having deeper knowledge of the nature of the missingness mechanisms will even enable us to obtain consistent estimators for datasets that are not MAR (in some cases).

### 5.1 INFORMED DELETION FOR MAR

Consider any MAR dataset, and a missingness graph where each  $R \in \mathbf{R}$  depends every observed variable in  $\mathbf{X}_o$ . This would be an MAR missingness graph that assumes

Table 2: Log-likelihoods of large networks, with higher treewidths, learned from MAR data (5 min. time limit).

| Size            |              | EM-JT   | EM-BP   | D-MCAR         | F-MCAR        | D-MAR         | F-MAR         |       | EM-JT   | EM-BP         | D-MCAR | F-MCAR        | D-MAR         | F-MAR         |
|-----------------|--------------|---------|---------|----------------|---------------|---------------|---------------|-------|---------|---------------|--------|---------------|---------------|---------------|
| 10 <sup>2</sup> | Grid 90-20-1 | -       | -57.14  | -80.92         | -57.01        | -80.80        | <b>-56.53</b> | Water | -19.10  | <b>-18.76</b> | -25.31 | -21.76        | -25.29        | -21.81        |
| 10 <sup>3</sup> |              | -       | -65.41  | -38.54         | -30.07        | -38.27        | <b>-29.86</b> |       | -       | <b>-14.73</b> | -19.13 | -16.45        | -18.93        | -16.36        |
| 10 <sup>4</sup> |              | -       | -       | -25.95         | -23.30        | -25.36        | <b>-22.88</b> |       | -       | -20.70        | -16.66 | -14.90        | -16.33        | <b>-14.67</b> |
| 10 <sup>5</sup> |              | -       | -       | -22.74         | -22.01        | <b>-21.60</b> | -             |       | -       | -             | -15.49 | -             | <b>-14.90</b> | -             |
| 10 <sup>2</sup> |              | Munin 1 | -       | <b>-103.72</b> | -115.50       | -105.81       | -115.41       |       | -104.87 | Barley        | -      | -89.22        | -89.54        | -89.26        |
| 10 <sup>3</sup> | -            |         | -69.03  | -71.01         | -65.91        | -70.61        | <b>-65.51</b> | -     | -74.26  |               | -71.67 | -70.46        | -71.68        | <b>-70.18</b> |
| 10 <sup>4</sup> | -            |         | -157.23 | -56.07         | <b>-54.24</b> | -55.46        | -             | -     | -       |               | -56.44 | <b>-55.12</b> | -56.40        | -             |
| 10 <sup>5</sup> | -            |         | -       | <b>-52.00</b>  | -             | -             | -             | -     | -       |               | -      | -             | -             | -             |

Table 3: Log-likelihoods of large networks, with higher treewidths, learned from MAR data (25 min. time limit).

| Size            |              | EM-JT   | EM-BP         | D-MCAR        | F-MCAR        | D-MAR   | F-MAR         |       | EM-JT   | EM-BP         | D-MCAR        | F-MCAR        | D-MAR         | F-MAR         |
|-----------------|--------------|---------|---------------|---------------|---------------|---------|---------------|-------|---------|---------------|---------------|---------------|---------------|---------------|
| 10 <sup>2</sup> | Grid 90-20-1 | -       | <b>-49.15</b> | -80.00        | -56.45        | -79.81  | -55.94        | Water | -18.88  | <b>-18.73</b> | -25.84        | -22.11        | -25.87        | -22.25        |
| 10 <sup>3</sup> |              | -       | -53.64        | -38.14        | -29.32        | -37.75  | <b>-29.09</b> |       | -17.63  | <b>-14.41</b> | -18.39        | -15.95        | -18.27        | -15.79        |
| 10 <sup>4</sup> |              | -       | -85.65        | -26.21        | -23.05        | -25.45  | <b>-22.62</b> |       | -       | -14.52        | -15.57        | -14.07        | -15.24        | <b>-13.92</b> |
| 10 <sup>5</sup> |              | -       | -             | -22.78        | -21.54        | -21.60  | <b>-20.79</b> |       | -       | -24.99        | -14.17        | -13.46        | -13.71        | <b>-13.19</b> |
| 10 <sup>6</sup> |              | -       | -             | -             | -             | -       | -             |       | -       | -             | <b>-13.73</b> | -             | -             | -             |
| 10 <sup>2</sup> |              | Munin 1 | -             | <b>-99.15</b> | -114.76       | -106.07 | -114.66       |       | -105.12 | Barley        | -89.05        | -89.15        | -89.57        | -89.17        |
| 10 <sup>3</sup> | -            |         | -67.85        | -74.18        | -67.81        | -73.82  | <b>-67.39</b> | -     | -70.38  |               | -71.86        | -70.54        | -71.87        | <b>-70.27</b> |
| 10 <sup>4</sup> | -            |         | -66.62        | -57.50        | -54.94        | -56.96  | <b>-54.64</b> | -     | -76.48  |               | -56.37        | <b>-55.13</b> | -56.33        | -             |
| 10 <sup>5</sup> | -            |         | -             | -53.07        | <b>-51.66</b> | -52.27  | -             | -     | -       |               | -51.31        | -             | <b>-51.19</b> | -             |

the least, in terms of conditional independencies, about the causal mechanisms  $\mathbf{R}$ . If we know more about the nature of the missingness (i.e., the variables that the  $\mathbf{R}$  depend on), we can exploit this to obtain more accurate estimates. Note that knowing the parents of an  $R$  is effectively equivalent to knowing the Markov blanket of  $R$  (Pearl, 1987), which can be learned from data (Tsamardinos, Aliferis, Statnikov, & Statnikov, 2003; Yaramakala & Margaritis, 2005). With sufficient domain knowledge, an expert may be able to specify the parents of the  $\mathbf{R}$ . It suffices even to identify a set of variables that just *contains* the Markov blanket.

Suppose that we have such knowledge of the missing data mechanisms of an MAR problem, namely that we know the subset  $\mathbf{W}_o$  of the observed variables  $\mathbf{X}_o$  that suffice to separate the missing values from their causal mechanisms, i.e., where  $\mathbf{X}_m \perp\!\!\!\perp \mathbf{R} \mid \mathbf{W}_o$ . We can exploit this knowledge in our direct deletion algorithm, to obtain improved parameter estimates. In particular, we can reduce the scope of the summation in our direct deletion algorithm from the variables  $\mathbf{X}'_o$  (the set of variables in  $\mathbf{X}_o$  that lie outside the family  $\mathbf{Y}$ ), to the variables  $\mathbf{W}'_o$  (the set of variables in  $\mathbf{W}_o$  that lie outside the family  $\mathbf{Y}$ ), yielding the algorithm:

$$\Pr(\mathbf{Y}) \approx \sum_{\mathbf{W}'_o} \Pr_{\mathcal{D}}(\mathbf{Y}_m^* | \mathbf{Y}_o, \mathbf{W}'_o, \mathbf{R}_{\mathbf{Y}_m} = \text{ob}) \Pr_{\mathcal{D}}(\mathbf{Y}_o, \mathbf{W}'_o)$$

Again, we need only consider, in the summation, the instantiations of  $\mathbf{W}'_o$  that appear in the dataset.

Table 4: alarm network with Informed MAR data

| Size                                     | F-MCAR        | D-MAR  | F-MAR  | ID-MAR        | IF-MAR        |
|--|---------------|--------|--------|---------------|---------------|
| Kullback-Leibler Divergence              |               |        |        |               |               |
| 10 <sup>2</sup>                          | <b>1.921</b>  | 2.365  | 2.364  | 2.021         | 2.011         |
| 10 <sup>3</sup>                          | 0.380         | 0.454  | 0.452  | 0.399         | <b>0.375</b>  |
| 10 <sup>4</sup>                          | 0.073         | 0.071  | 0.072  | 0.059         | <b>0.053</b>  |
| 10 <sup>5</sup>                          | 0.041         | 0.021  | 0.022  | 0.011         | <b>0.010</b>  |
| 10 <sup>6</sup>                          | 0.040         | 0.006  | 0.008  | <b>0.001</b>  | <b>0.001</b>  |
| Test Set Log-Likelihood (Fully Observed) |               |        |        |               |               |
| 10 <sup>2</sup>                          | <b>-11.67</b> | -12.13 | -12.13 | -11.77        | -11.76        |
| 10 <sup>3</sup>                          | <b>-10.40</b> | -10.47 | -10.47 | -10.42        | <b>-10.40</b> |
| 10 <sup>4</sup>                          | -10.04        | -10.04 | -10.04 | <b>-10.02</b> | <b>-10.02</b> |
| 10 <sup>5</sup>                          | -10.00        | -9.98  | -9.98  | <b>-9.97</b>  | <b>-9.97</b>  |
| 10 <sup>6</sup>                          | -10.00        | -9.97  | -9.97  | <b>-9.96</b>  | <b>-9.96</b>  |

We refer to this algorithm as *informed direct deletion*. By reducing the scope of the summation, we need to estimate fewer sub-terms  $\Pr_{\mathcal{D}}(\mathbf{Y}_m^* | \mathbf{Y}_o, \mathbf{W}'_o, \mathbf{R}_{\mathbf{Y}_m} = \text{ob})$ . This results in a more efficient computation, but further, each individual sub-expression can be estimated on more data. Moreover, our estimates remain consistent. We can similarly replace  $\mathbf{X}_o$  by  $\mathbf{W}_o$  in the factored deletion algorithm, to obtain an *informed factored deletion* algorithm.

**Empirical Evaluation** Here, we evaluate the benefits of informed deletion. In addition to the MAR assumption, with this setting, we assume that we know the set of parents  $\mathbf{W}_o$  of the missingness mechanism variables. To gen-

erate data for such a mechanism, we select a random set of  $s$  variables to form  $\mathbf{W}_o$ . We further employ the sampling algorithm previously used for MAR data, but now insist that the parents of  $\mathbf{R}$  variables come from  $\mathbf{W}_o$ . Table 4 shows likelihoods and KLDs on the `alarm` network, for  $s = 3$ , and other settings as in the MAR experiments. Informed D-MAR (ID-MAR) and F-MAR (IF-MAR) consistently outperform their non-informed counterparts.

## 5.2 LEARNING FROM MNAR DATA

A missing data problem that is not MAR is classified as MNAR. Here, the parameters of a Bayesian network may not even be identifiable. Further, maximum-likelihood estimation is in general not consistent, so EM and gradient methods can yield biased estimates. However, if one knows the mechanisms that dictate missingness (in the form of a missingness graph), it becomes possible again to obtain consistent estimates, in some cases (Mohan et al., 2013).

For example, consider the missingness graph of Figure 2, which is an MNAR problem, where both variables  $X$  and  $Y$  are partially observed, and the missingness of each variable depends on the value of the other. Here, it is still possible to obtain consistent parameter estimates, as  $\Pr(X, Y) =$

$$\frac{\Pr(R_X = \text{ob}, R_Y = \text{ob}) \Pr(X^*, Y^* | R_X = \text{ob}, R_Y = \text{ob})}{\Pr(R_X = \text{ob} | Y^*, R_Y = \text{ob}) \Pr(R_Y = \text{ob} | X^*, R_X = \text{ob})}$$

For a derivation, see Mohan et al. (2013). Such derivations for recovering queries under MNAR are extremely sensitive to the structure of the missingness graph. Indeed, the class of missingness graphs that admit consistent estimation has not yet been fully characterized.

## 6 RELATED WORK

When estimating the parameters of a Bayesian network, maximum-likelihood (ML) estimation is the typical approach, where for incomplete data, the common wisdom among machine learning practitioners is that one needs to use Expectation-Maximization (EM) or gradient methods (Dempster, Laird, & Rubin, 1977; Lauritzen, 1995); see also, e.g., Darwiche (2009), Koller and Friedman (2009), Murphy (2012), Barber (2012). Again, such methods do not scale to large datasets or large networks as (1) they are iterative, (2) they suffer from local optima, and most notably, (3) they require inference in a Bayesian network. Considerable effort has been expended in improving on EM across these dimensions, in order to, for example, (1) accelerate the convergence of EM, and to intelligently sample subsets of a dataset, e.g., Thiesson et al. (2001), (2) escape local optima, e.g., (Elidan, Ninio, Friedman, & Shuurmans, 2002), and (3) use approximate inference algorithms in lieu of exact ones when inference is intractable, e.g., Ghahramani and Jordan (1997), Caffo, Jank, and Jones

(2005). Further, while EM is suitable for data that is MAR (the typical assumption in practice), there are some exceptions, such as work on recommender systems that explicitly incorporate missing data mechanisms (Marlin & Zemel, 2009; Marlin, Zemel, Roweis, & Slaney, 2007, 2011).

In the case of complete data, the parameter estimation task simplifies considerably, in the case of Bayesian networks: maximum-likelihood estimates can be obtained inference-free and in closed-form, using just a single pass over the data:  $\theta_{x|\mathbf{u}} = \Pr_{\mathcal{D}}(x|\mathbf{u})$ . In fact, the estimation algorithms that we proposed in this paper also obtain the same parameter estimates in the case of complete data, although we are not concerned with maximum-likelihood estimation here—we simply want to obtain estimates that are consistent (as in estimation by the method of moments).

Other inference-free estimators have been proposed for other classes of graphical models. Abbeel, Koller, and Ng (2006) identified a method for closed-form, inference-free parameter estimation in factor graphs of bounded degree from complete data. More recently, Halpern and Sontag (2013) proposed an efficient, inference-free method for consistently estimating the parameters of noisy-or networks with latent variables, under certain structural assumptions. From the perspective of maximum-likelihood learning, where evaluating the likelihood (requiring inference) seems to be unavoidable, the ability to consistently estimate parameters—without the need for inference—greatly extends the accessibility and utility of such models. For example, it opens the door to practical structure learning algorithms, under incomplete data, which is a notoriously difficult problem in practice (Abbeel et al., 2006; Jernite, Halpern, & Sontag, 2013).

## 7 CONCLUSIONS

In summary, we proposed a family of efficient and scalable algorithms for learning the parameters of Bayesian networks, from MCAR and MAR datasets, and sometimes MNAR datasets. Our parameter estimates are asymptotically consistent, and further, they are obtained inference-free and in closed-form. We further introduced and discussed some improved approaches for parameter estimation, when given additional knowledge of the missingness mechanisms underlying an incomplete dataset. Empirically, we demonstrate the practicality of our method, showing that it can scale to much larger datasets, and much larger Bayesian networks, than EM.

### Acknowledgments

This work was supported in part by ONR grants #N00014-10-1-0933, #N00014-12-1-0423 and #N00014-13-1-0153, by NSF grants #IIS-1118122 and #IIS-1302448, and the Research Foundation-Flanders (FWO-Vlaanderen).

## References

- Abbeel, P., Koller, D., & Ng, A. Y. (2006). Learning factor graphs in polynomial time and sample complexity. *Journal of Machine Learning Research*, 7, 1743–1788.
- Barber, D. (2012). *Bayesian Reasoning and Machine Learning*. Cambridge University Press.
- Caffo, B. S., Jank, W., & Jones, G. L. (2005). Ascent-based monte carlo expectation-maximization. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 67(2), pp. 235–251.
- Chavira, M., & Darwiche, A. (2006). Encoding CNFs to empower component analysis. In *SAT*, pp. 61–74.
- Chavira, M., & Darwiche, A. (2007). Compiling Bayesian networks using variable elimination. In *Proceedings of IJCAI*, pp. 2443–2449.
- Darwiche, A. (2009). *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press.
- Dempster, A., Laird, N., & Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39, 1–38.
- Elidan, G., Ninio, M., Friedman, N., & Shuurmans, D. (2002). Data perturbation for escaping local maxima in learning. In *Proceedings of AAAI*, pp. 132–139.
- Francois, O., & Leray, P. (2007). Generation of incomplete test-data using Bayesian networks. In *IJCNN*, pp. 2391–2396.
- Ghahramani, Z., & Jordan, M. I. (1997). Factorial hidden markov models. *Machine Learning*, 29(2-3), 245–273.
- Halpern, Y., & Sontag, D. (2013). Unsupervised learning of noisy-or Bayesian networks. In *Proceedings of UAI*.
- Jernite, Y., Halpern, Y., & Sontag, D. (2013). Discovering hidden variables in noisy-or networks using quartet tests. In *NIPS*, pp. 2355–2363.
- Koller, D., & Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques*. MIT Press.
- Lauritzen, S. (1995). The EM algorithm for graphical association models with missing data. *Computational Statistics and Data Analysis*, 19, 191–201.
- Marlin, B., & Zemel, R. (2009). Collaborative prediction and ranking with non-random missing data. In *Proceedings of the third ACM conference on Recommender systems*, pp. 5–12. ACM.
- Marlin, B., Zemel, R., Roweis, S., & Slaney, M. (2007). Collaborative filtering and the missing at random assumption. In *UAI*.
- Marlin, B., Zemel, R., Roweis, S., & Slaney, M. (2011). Recommender systems: missing data and statistical model estimation. In *IJCAI*.
- Mohan, K., Pearl, J., & Tian, J. (2013). Graphical models for inference with missing data. In *Proceedings of NIPS*.
- Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press.
- Pearl, J. (1987). Evidential reasoning using stochastic simulation of causal models. *AIJ*, 32(2), 245–257.
- Rubin, D. B. (1976). Inference and missing data. *Biometrika*, 63(3), 581–592.
- Thiesson, B., Meek, C., & Heckerman, D. (2001). Accelerating EM for large databases. *Machine Learning*, 45(3), 279–299.
- Thoemmes, F., & Mohan, K. (2015). Graphical representation of missing data problems. *Structural Equation Modeling: A Multidisciplinary Journal*. To appear.
- Tsamardinos, I., Aliferis, C. F., Statnikov, A. R., & Statnikov, E. (2003). Algorithms for large scale Markov blanket discovery.. In *Proceedings of FLAIRS*, Vol. 2003, pp. 376–381.
- Wasserman, L. (2011). *All of Statistics*. Springer Science & Business Media.
- Yaramakala, S., & Margaritis, D. (2005). Speculative markov blanket discovery for optimal feature selection. In *Proceedings of ICDM*.

---

# Bayes Optimal Feature Selection for Supervised Learning with General Performance Measures

---

**Saneem Ahmed C.G.**<sup>†</sup>  
IBM Research  
Bangalore 560045, India

**Harikrishna Narasimhan**  
Indian Institute of Science  
Bangalore 560012, India

**Shivani Agarwal**  
Indian Institute of Science  
Bangalore 560012, India

## Abstract

The problem of feature selection is critical in several areas of machine learning and data analysis. Here we consider feature selection for supervised learning problems, where one wishes to select a small set of features that facilitate learning a good prediction model in the reduced feature space. Our interest is primarily in filter methods that select features independently of the learning algorithm to be used and are generally faster to implement than wrapper methods. Many common filter methods for feature selection make use of mutual information based criteria to guide their search process. However, even in simple binary classification problems, mutual information based methods do not always select the best set of features in terms of the Bayes error. In this paper, we develop a filter method that directly aims to select the optimal set of features for a general performance measure of interest. Our approach uses the Bayes error with respect to the given performance measure as the criterion for feature selection and applies a greedy algorithm to optimize this criterion. We demonstrate application of this method to a variety of learning problems involving different performance measures. Experiments suggest the proposed approach is competitive with several state-of-the-art methods.

## 1 INTRODUCTION

The problem of feature selection is critical in several areas of machine learning and data analysis, particularly for learning prediction models with good generalization ability and for reducing the running time of learning algorithms [1–3]. In this paper, we consider feature selection for supervised learning problems, where one wishes to select a small set of features that facilitate learning a good

prediction model in the reduced feature space. Our focus is primarily on filter methods that select features independently of the learning algorithm to be used, typically by greedily maximizing some suitable feature selection criterion. These methods are generally faster in practice and easier to implement than other approaches to feature selection such as wrapper or embedded methods.

Over the years, there has been much work on designing filter methods for feature selection, many of which make use of mutual information based criteria to guide their search process [4–7]. However, these methods do not explicitly consider the performance measure used to evaluate a model in the learning problem. In fact, even in the case of simple binary classification, one can construct settings where the popular mutual information criterion does not yield the best set of features in terms of the Bayes 0-1 classification error (as we shall shortly see with an example) [8]. Clearly, there is a need for filter methods that are tailored to directly optimize a given performance measure of interest.

In this paper, we develop a *Bayes optimal* filter method for a general performance measure. Our approach directly aims to find the optimal set of features in terms of the Bayes error for the given loss or performance measure, thus allowing for the possibility of learning a good model in the reduced feature space. We show that the mutual information criterion mentioned above is a special case of our setting when the loss function of interest is the logarithmic loss for class probability estimation. We use a greedy forward selection algorithm for approximately optimizing the Bayes criterion for the given performance measure, and demonstrate application of this method to various learning problems involving different performance measures. Experiments on several learning tasks suggest that the proposed approach is competitive with the state-of-the-art methods.

Indeed in the simpler setting of classification with the 0-1 error, there have been some works that have suggested the use of Bayes error as a criterion for feature selection/transformation [8–13]. Of these, only Yang and Hu (2012) provide an experimental evaluation of a filter method for optimizing the Bayes 0-1 error [13]; however,

---

<sup>†</sup>Work done while at Indian Institute of Science, Bangalore.

even here, the objective eventually optimized is different from the Bayes optimal criterion for the 0-1 error (we elaborate on this in Section 3.1). On the other hand, we provide in this paper the first systematic study of Bayes optimal filter methods for general performance measures, going well beyond the simple setting of 0-1 classification, and handling a variety of learning settings, including those with complex performance measures such as the F-measure.

## 1.1 RELATED WORK

Filter methods have received much attention from the machine learning/data mining/artificial intelligence communities, resulting in various hand-crafted filter criteria and heuristic techniques for optimizing the proposed objectives [4–7, 14–21]. Predominant among these are methods that use the mutual information (MI) between a given feature subset and the output label as a measure of *relevance* of the feature subset to the given learning task, often with additional information theoretic terms to account for *redundancy* among the features in the given subset [4–7, 16–21]. While there have been arguments made to justify the use of MI as a criterion for binary classification by establishing lower/upper bounds on MI in terms of the 0-1 Bayes error [21–24], these bounds are tight only for certain settings; in general, the optimal feature subset for the MI criterion need not be the same as that for the 0-1 Bayes error.

There has also been some work on designing filter methods for specific learning tasks, such as text retrieval [25], class imbalanced classification [26], and ranking [27]. However, the feature selection criteria proposed therein are either based on heuristics and do not explicitly promote feature subsets that are Bayes optimal for the given problem, or as in the case of [25], require a certain (binary) representation of the features and do not apply to general settings.

Apart from filter methods, other popular families of feature selection techniques include *wrapper methods*, where the quality of a subset of features is determined by explicitly learning a model on the feature subset and evaluating its accuracy on a held-out sample [2, 28, 29]; and *embedded methods*, which combine model learning and feature selection into a single step, such as using sparse regularization in the learning problem [30]. While both these approaches allow us to incorporate different loss functions during feature selection, filter methods are computationally cheaper as they decouple feature selection from model learning, and are typically simpler to implement in practice.

**Organization.** Section 2 gives preliminaries, together with an example illustrating that the MI feature selection criterion can be suboptimal for binary classification. Section 3 describes the proposed Bayes feature selection method, followed by examples of how it can be applied to different learning problems and performance measures. Section 4 gives results of experiments on several learning tasks.

## 2 PRELIMINARIES AND BACKGROUND

**Notation.** For  $n \in \mathbb{Z}_+$ , we denote  $[n] = \{1, \dots, n\}$ . For a vector  $\mathbf{x} = (x_1, \dots, x_n) \in \mathbb{R}^n$  and set  $\mathcal{J} = \{j_1, \dots, j_k\} \subseteq [n]$  with  $j_1 < \dots < j_k$ , we denote  $\mathbf{x}_{\mathcal{J}} = (x_{j_1}, \dots, x_{j_k}) \in \mathbb{R}^k$ . For random variables  $X$  and  $Y$ , we denote by  $H(X)$  the entropy of  $X$ , by  $H(Y|X)$  the conditional entropy of  $Y$  given  $X$ , and by  $I(X; Y)$  the mutual information between  $X$  and  $Y$ . For a predicate  $\phi$ , we denote by  $\mathbf{1}(\phi)$  the indicator of  $\phi$ , which takes the value 1 if  $\phi$  is true and 0 otherwise. For any  $z \in \mathbb{R}$ ,  $\text{sign}(z) = 1$  if  $z > 0$  and  $-1$  otherwise.

**Problem Setup.** Let  $\mathcal{X} \subseteq \mathbb{R}^n$  be an  $n$ -dimensional instance space. We will be interested in feature selection for *supervised learning problems*, where there is some label space  $\mathcal{Y}$  and prediction space  $\hat{\mathcal{Y}}$ ; one receives a training sample  $S = ((\mathbf{x}^1, y^1), \dots, (\mathbf{x}^m, y^m)) \in (\mathcal{X} \times \mathcal{Y})^m$ , and the goal is to learn a prediction model  $h : \mathcal{X} \rightarrow \hat{\mathcal{Y}}$ .<sup>1</sup> Typically, one assumes all examples (both training examples and future test examples) are drawn i.i.d. from some probability distribution  $D$  on  $\mathcal{X} \times \mathcal{Y}$ , and the goal is to learn a prediction model with good prediction performance (according to a suitable performance measure) on future examples from  $D$ . We will denote by  $(X, Y)$  a random variable drawn from  $D$ . Often, performance is measured via a loss function  $\ell : \mathcal{Y} \times \hat{\mathcal{Y}} \rightarrow \mathbb{R}_+$ ; the goal then is to learn a model  $h$  minimizing the expected loss on a new example from  $D$ , which we refer to as the  $\ell$ -error of  $h$  w.r.t.  $D$ :  $\text{er}_D^\ell[h] = \mathbf{E}_{(X, Y) \sim D}[\ell(Y, h(X))]$ . The smallest achievable  $\ell$ -error over all possible prediction models is called the *Bayes  $\ell$ -error* for  $D$ :  $\text{er}_D^{\ell, *} = \inf_{h: \mathcal{X} \rightarrow \hat{\mathcal{Y}}} \text{er}_D^\ell[h]$ . For example, in binary classification, one has  $\mathcal{Y} = \hat{\mathcal{Y}} = \{\pm 1\}$ , and the loss function of interest is often the 0-1 loss  $\ell_{0-1} : \{\pm 1\} \times \{\pm 1\} \rightarrow \mathbb{R}_+$  defined as  $\ell_{0-1}(y, \hat{y}) = \mathbf{1}(\hat{y} \neq y)$ . For problems with binary labels  $\mathcal{Y} = \{\pm 1\}$ , we will denote by  $p = \mathbf{P}(Y = 1)$  the overall probability of label +1 under  $D$ , and by  $\eta : \mathcal{X} \rightarrow [0, 1]$  the associated class probability function:  $\eta(\mathbf{x}) = \mathbf{P}(Y = 1 | X = \mathbf{x})$ . Here the Bayes 0-1 error has the form  $\text{er}_D^{0-1, *} = \mathbf{E}_X[\min(\eta(X), 1 - \eta(X))]$ .

The *feature selection problem* we are interested in is to select a subset of features  $\mathcal{J} \subseteq [n]$  of some specified size  $k \in [n]$  (usually  $k \ll n$ ), such that one can then learn a good prediction model in the reduced  $k$ -dimensional feature space  $\mathcal{X}_{\mathcal{J}} = \{\mathbf{x}_{\mathcal{J}} | \mathbf{x} \in \mathcal{X}\} \subseteq \mathbb{R}^k$ .<sup>2</sup> Specifically, given a training sample  $S \in (\mathcal{X} \times \mathcal{Y})^m$  as above, one works with the reduced training sample  $S_{\mathcal{J}} = ((\mathbf{x}_{\mathcal{J}}^1, y^1), \dots, (\mathbf{x}_{\mathcal{J}}^m, y^m)) \in (\mathcal{X}_{\mathcal{J}} \times \mathcal{Y})^m$ , and learns a prediction model  $h_{\mathcal{J}} : \mathcal{X}_{\mathcal{J}} \rightarrow \hat{\mathcal{Y}}$  in the reduced space  $\mathcal{X}_{\mathcal{J}}$ . We will denote by  $D_{\mathcal{J}}$  the marginal distribution of  $D$  on  $\mathcal{X}_{\mathcal{J}} \times \mathcal{Y}$ ; for problems with binary la-

<sup>1</sup>Often  $\hat{\mathcal{Y}} = \mathcal{Y}$ , but this is not always the case.

<sup>2</sup>In this paper, we assume for simplicity that the target feature subset size  $k$  is given as part of the problem. However, the methods developed easily extend to settings where  $k$  is unknown.

bels  $\mathcal{Y} = \{\pm 1\}$ , we will also denote by  $\eta_{\mathcal{J}} : \mathcal{X}_{\mathcal{J}} \rightarrow [0, 1]$  the class probability function on the reduced feature space  $\mathcal{X}_{\mathcal{J}}$ :  $\eta_{\mathcal{J}}(\mathbf{z}) = \mathbf{P}(Y = 1 | X_{\mathcal{J}} = \mathbf{z})$ , where  $X_{\mathcal{J}}$  contains components of the random vector  $X$  corresponding to indices in  $\mathcal{J}$ . Clearly, if the examples in  $S$  are drawn i.i.d. from  $D$ , then the examples in  $S_{\mathcal{J}}$  can be viewed as being drawn i.i.d. from  $D_{\mathcal{J}}$ . In the loss function setting, the performance of a model  $h_{\mathcal{J}}$  learned in the reduced feature space is measured via its  $\ell$ -error w.r.t.  $D_{\mathcal{J}}$ :  $\text{er}_{D_{\mathcal{J}}}^{\ell}[h_{\mathcal{J}}] = \mathbf{E}_{(Z, Y) \sim D_{\mathcal{J}}}[\ell(Y, h_{\mathcal{J}}(Z))] = \mathbf{E}_{(X, Y) \sim D}[\ell(Y, h_{\mathcal{J}}(X_{\mathcal{J}}))]$ .

**Feature Selection as (Approximate) Optimization.** We will view feature selection methods as (approximately) optimizing some objective or criterion  $C_D : 2^{[n]} \rightarrow \mathbb{R}$ , which typically depends on distribution  $D$ . Given such a criterion  $C_D$  and a target feature subset size  $k$ , one aims to select

$$\mathcal{J}^* \in \underset{\substack{\mathcal{J} \subseteq [n] \\ |\mathcal{J}| = k}}{\text{argmax}} C_D(\mathcal{J}). \quad (1)$$

Of course, in practice, one does not know the distribution  $D$ , and so instead uses an approximate version of the criterion  $C_D$  based on the training sample  $S$ , which we shall denote as  $\widehat{C}_S : 2^{[n]} \rightarrow \mathbb{R}$ . Moreover, the combinatorial optimization problem (over  $\binom{n}{k}$  subsets) is generally computationally hard, and so one settles for an approximate search strategy, such as a greedy approach. We shall elaborate further on both these approximations below.

**Filter Methods and Mutual Information (MI) Criterion.** In a filter method for feature selection, the choice of the feature subset does not depend on the particular learning algorithm to be used in the reduced feature space, i.e. the criterion  $C_D$  is independent of the particular learning algorithm to be used. A popular filter criterion that is widely used in feature selection for supervised learning is the *mutual information* (MI) criterion, defined as the mutual information between the selected features and the labels:

$$C_D^{\text{MI}}(\mathcal{J}) = I(X_{\mathcal{J}}; Y), \quad (2)$$

where  $(X, Y)$  denotes a random variable distributed according to  $D$ . The motivation for using the MI criterion is that it is expected to preserve the information necessary for learning a good prediction model. Indeed, in the case of binary classification, monotonic functions of the mutual information  $I(X; Y)$  are known to both upper and lower bound the Bayes error  $\text{er}_D^{0-1,*}$  [21–24]. However, as seen below, even in the case of binary classification, there are situations where the MI criterion does *not* select an optimal set of features:

**Example 1** (Suboptimality of MI criterion for binary classification with 0-1 error). *Consider a binary classification problem on a 2-dimensional instance space with binary features:  $\mathcal{X} = \{0, 1\}^2$ ,  $\mathcal{Y} = \{\pm 1\}$ . Let  $D$  be a probability distribution on  $(\mathcal{X} \times \mathcal{Y})$  under which  $\mathbf{P}(Y = 1) = 0.3$ , the random variables  $X_1, X_2$  (components of the random*

*vector  $X$ ) are conditionally independent given the label  $Y$ , and the class-conditional distributions are given by*

$$\begin{aligned} \mathbf{P}(X_1 = 1 | Y = 1) &= 0.4; & \mathbf{P}(X_1 = 1 | Y = -1) &= 0.1; \\ \mathbf{P}(X_2 = 1 | Y = 1) &= 0.9; & \mathbf{P}(X_2 = 1 | Y = -1) &= 0.4. \end{aligned}$$

*Clearly,  $\mathbf{P}(X_1 = 1) = 0.19$ ,  $\mathbf{P}(X_2 = 1) = 0.55$ ,  $\eta_{\{1\}}(0) = 0.22$ ,  $\eta_{\{1\}}(1) = 0.63$ ,  $\eta_{\{2\}}(0) = 0.07$  and  $\eta_{\{2\}}(1) = 0.49$ . Now consider selecting a single feature for use in learning a binary classifier (thus here  $n = 2$ ,  $k = 1$ ). It can be verified that under the above distribution,*

$$\begin{aligned} C_D^{\text{MI}}(\{1\}) &= I(X_1; Y) = 0.08 \\ C_D^{\text{MI}}(\{2\}) &= I(X_2; Y) = 0.17. \end{aligned}$$

*Therefore the MI criterion would select feature 2 and learn a classifier in the feature space  $\mathcal{X}_{\{2\}}$ . One can also compute the Bayes 0-1 errors in  $\mathcal{X}_{\{1\}}$  and  $\mathcal{X}_{\{2\}}$ ; these can be verified to be*

$$\text{er}_{D_{\{1\}}}^{0-1,*} = 0.25; \quad \text{er}_{D_{\{2\}}}^{0-1,*} = 0.30.$$

*Thus even if one uses the best possible learning algorithm in the feature space  $\mathcal{X}_{\{2\}}$  selected by the MI criterion, the best classifier one can learn will have 0-1 error 0.30. On the other hand, if we had selected feature 1, we could potentially have learned a classifier with 0-1 error 0.25!*

The above example suggests looking directly for a feature subset that yields low Bayes error with respect to a given performance measure of interest.

### 3 BAYES OPTIMAL FEATURE SELECTION

Motivated by the above discussion, we now develop a filter method for feature selection that is tailored to optimize a general performance measure of interest. In particular, rather than selecting a feature subset by maximizing the mutual information with the labels, our approach optimizes *the information most relevant to the supervised learning task at hand*, with the aim of learning as good a prediction model in the reduced feature space as possible in terms of the given loss or performance measure. More formally, for a supervised learning problem with label space  $\mathcal{Y}$ , prediction space  $\widehat{\mathcal{Y}}$ , and with loss function  $\ell : \mathcal{Y} \times \widehat{\mathcal{Y}} \rightarrow \mathbb{R}_+$ , we will be interested in selecting a feature subset that minimizes the Bayes  $\ell$ -error in the reduced feature space, or equivalently, maximizes the following criterion:

$$C_D^{\text{Bayes}, \ell}(\mathcal{J}) = -\text{er}_{D_{\mathcal{J}}}^{\ell,*}. \quad (3)$$

Note that this is different from a wrapper method, which looks for a feature subset that maximizes prediction performance of a model learned by a particular algorithm; here, we are instead interested in finding the best feature subset for a given performance measure of interest, *without being tied to any particular learning algorithm*.

### 3.1 EXAMPLES OF BAYES CRITERION FOR VARIOUS LEARNING PROBLEMS AND PERFORMANCE MEASURES

Here we give several examples of the above Bayes criterion for specific learning problems/performance measures. We shall see that for the case of binary class probability estimation with the logarithmic loss, the Bayes criterion effectively reduces to the MI criterion (Example 5); thus the MI criterion can be viewed as finding a good feature space for class probability estimation. Similarly, for regression with squared error, the Bayes criterion is exactly the criterion optimized in the forward regression feature selection algorithm for sparse linear regression (Example 6). We begin with the simple case of binary classification with 0-1 error.

**Example 2** (Bayes criterion for binary classification with 0-1 error). *Let  $\mathcal{Y} = \hat{\mathcal{Y}} = \{\pm 1\}$ , with  $\ell_{0-1} : \{\pm 1\} \times \{\pm 1\} \rightarrow \mathbb{R}_+$  defined as  $\ell_{0-1}(y, \hat{y}) = \mathbf{1}(\hat{y} \neq y)$ . Then*

$$C_D^{\text{Bayes}, 0-1}(\mathcal{J}) = -\mathbf{E}_X \left[ \min(\eta_{\mathcal{J}}(X_{\mathcal{J}}), 1 - \eta_{\mathcal{J}}(X_{\mathcal{J}})) \right].$$

As noted earlier, the filter method provided by Yang and Hu (2012) [13] for optimizing the Bayes 0-1 error eventually optimizes an objective different from the above one; while the authors initially discuss a feature selection criterion of the above form, they end up prescribing and analyzing a variant  $-\mathbf{E}_{X,Y} [(1-Y)\eta_{\mathcal{J}}(X_{\mathcal{J}}) + Y(1-\eta_{\mathcal{J}}(X_{\mathcal{J}}))] = -\mathbf{E}_X [2\eta_{\mathcal{J}}(X_{\mathcal{J}})(1-\eta_{\mathcal{J}}(X_{\mathcal{J}}))]$ , which is not necessarily optimal for the 0-1 error (see Eq. (7) in their paper). In this work, we go well beyond the simple setting of 0-1 classification, and present a systematic study of Bayes optimal criteria for general performance measures, as seen below.

**Example 3** (Bayes criterion for binary classification with cost-sensitive error). *Let  $\mathcal{Y} = \hat{\mathcal{Y}} = \{\pm 1\}$ . Let  $c \in (0, 1)$  denote the cost of a false positive and  $(1-c)$  the cost of a false negative; the corresponding cost-sensitive loss  $\ell_c : \{\pm 1\} \times \{\pm 1\} \rightarrow \mathbb{R}_+$  is defined as*

$$\ell_c(y, \hat{y}) = \begin{cases} c & \text{if } y = -1, \hat{y} = 1 \\ 1-c & \text{if } y = 1, \hat{y} = -1 \\ 0 & \text{otherwise.} \end{cases}$$

Then

$$C_D^{\text{Bayes}, c}(\mathcal{J}) = -\mathbf{E}_X \left[ \min((1-c)\eta_{\mathcal{J}}(X_{\mathcal{J}}), c(1-\eta_{\mathcal{J}}(X_{\mathcal{J}}))) \right].$$

**Example 4** (Bayes criterion for binary classification with balanced 0-1 error). *Let  $\mathcal{Y} = \hat{\mathcal{Y}} = \{\pm 1\}$ . The balanced loss  $\ell_{\text{bal}} : \{\pm 1\} \times \{\pm 1\} \rightarrow \mathbb{R}_+$  seeks to balance prediction errors on positive and negative examples by weighting them according to their inverse class probabilities, and is frequently used to measure classification performance in class imbalance settings [31]; it depends on the underlying distribution  $D$  via the probability  $p = \mathbf{P}(Y = 1)$ , and*

is defined as

$$\ell_{\text{bal}}(y, \hat{y}) = \begin{cases} \frac{1}{1-p} & \text{if } y = -1, \hat{y} = 1 \\ \frac{1}{p} & \text{if } y = 1, \hat{y} = -1 \\ 0 & \text{otherwise.} \end{cases}$$

Here the Bayes criterion becomes

$$C_D^{\text{Bayes}, \text{bal}}(\mathcal{J}) = -\mathbf{E}_X \left[ \min\left(\frac{\eta_{\mathcal{J}}(X_{\mathcal{J}})}{p}, \frac{1-\eta_{\mathcal{J}}(X_{\mathcal{J}})}{1-p}\right) \right].$$

**Example 5** (Bayes criterion for binary class probability estimation with logarithmic loss). *Let  $\mathcal{Y} = \{\pm 1\}$  and  $\hat{\mathcal{Y}} = [0, 1]$ , with logarithmic loss  $\ell_{\log} : \{\pm 1\} \times [0, 1] \rightarrow \mathbb{R}_+$  defined as*

$$\ell_{\log}(y, \hat{y}) = -\mathbf{1}(y = 1) \ln(\hat{y}) - \mathbf{1}(y = -1) \ln(1 - \hat{y}).$$

Then

$$\begin{aligned} C_D^{\text{Bayes}, \log}(\mathcal{J}) &= -\mathbf{E}_X \left[ -\eta_{\mathcal{J}}(X_{\mathcal{J}}) \ln(\eta_{\mathcal{J}}(X_{\mathcal{J}})) \right. \\ &\quad \left. - (1 - \eta_{\mathcal{J}}(X_{\mathcal{J}})) \ln(1 - \eta_{\mathcal{J}}(X_{\mathcal{J}})) \right] \\ &= -H(Y|X_{\mathcal{J}}) = I(X_{\mathcal{J}}; Y) - H(Y) \\ &= C_D^{\text{MI}}(\mathcal{J}) - H(Y). \end{aligned}$$

*This is equivalent to using the MI criterion! Thus, in the binary setting, the MI criterion effectively selects a feature set that minimizes Bayes log-error, i.e. that allows for a good class probability estimator (in terms of logarithmic loss) in the resulting feature space! (Note that this is not the same as selecting good features for binary classification with 0-1 error or other performance measures; e.g. see Example 1. This is also demonstrated in our experiments in Section 4.)*

**Example 6** (Bayes criterion for regression with squared error). *Let  $\mathcal{Y} = \hat{\mathcal{Y}} = \mathbb{R}$ , with squared loss  $\ell_{\text{sq}} : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}_+$  defined as  $\ell_{\text{sq}}(y, \hat{y}) = (\hat{y} - y)^2$ . Then*

$$C_D^{\text{Bayes}, \text{sq}}(\mathcal{J}) = -\mathbf{E}_X [\text{Var}(Y | X_{\mathcal{J}})].$$

*This is exactly the criterion used in the well-known forward regression feature selection algorithm for sparse linear regression (where one assumes  $\mathbf{E}[Y|X = \mathbf{x}] = \beta^T \mathbf{x}$  for some sparse  $\beta \in \mathbb{R}^n$ ) [32].*

While all examples seen so far have involved performance measures that can be expressed as an expected value of a loss function, we shall next consider examples of learning problems where the performance measure of interest is complex and non-additive.

**Example 7** (Bayes criterion for binary classification/retrieval with  $F_{\beta}$ -measure). *Let  $\mathcal{Y} = \hat{\mathcal{Y}} = \{\pm 1\}$ , and consider a classification or retrieval problem where the goal is to learn a classifier  $h : \mathcal{X} \rightarrow \{\pm 1\}$  with performance measured by the  $F_{\beta}$ -measure (higher values are better):*

$$F_{\beta, D}[h] = \frac{1 + \beta^2}{\frac{\beta^2}{\text{Prec}_D[h]} + \frac{1}{\text{Rec}_D[h]}},$$



---

**Algorithm 1**  $\ell$ -BayesGreedy

---

- 1: **Inputs:**  $S = (\mathbf{x}^1, y^1), \dots, (\mathbf{x}^m, y^m) \in (\mathbb{R}^n \times \mathcal{Y})^m$   
 $k \in [n]$
  - 2: **Initialize:**  $\mathcal{J} \leftarrow \emptyset$
  - 3: **for**  $t = 1 \dots k$  **do**
  - 4:      $j_t \leftarrow \operatorname{argmax}_{j \in [n] \setminus \mathcal{J}} \widehat{C}_S^{\text{Bayes}, \ell}(\mathcal{J} \cup \{j\})$
  - 5:      $\mathcal{J} \leftarrow \mathcal{J} \cup \{j_t\}$
  - 6: **end for**
  - 7: **Output:**  $\mathcal{J}$
- 

where  $\text{Prec}_D[h] = \mathbf{P}(Y = 1 | h(X) = 1)$  and  $\text{Rec}_D[h] = \mathbf{P}(h(X) = 1 | Y = 1)$  are the precision and recall of  $h$ , respectively, and  $\beta > 0$  trades off the relative importance of these two quantities. In this case, the performance measure cannot be expressed as the expected value of a loss function over individual data points. Nevertheless, it is known that the Bayes optimal classifier for the  $F_\beta$ -measure is obtained by thresholding the class probability function  $\eta$  for the given distribution at an optimal point [24, 33]. One can therefore compute the Bayes optimal value of the  $F_\beta$ -measure for a given distribution, and use this as the criterion to be optimized in feature selection:

$$\begin{aligned} C_D^{\text{Bayes}, F_\beta}(\mathcal{J}) &= \sup_{h_{\mathcal{J}}: \mathcal{X}_{\mathcal{J}} \rightarrow \{\pm 1\}} F_{\beta, D_{\mathcal{J}}}[h_{\mathcal{J}}] \\ &= \sup_{t \in [0, 1]} F_{\beta, D_{\mathcal{J}}}[\text{sign} \circ (\eta_{\mathcal{J}} - t)]. \end{aligned}$$

**Example 8** (Bayes criterion for bipartite ranking with AUC). Let  $\mathcal{Y} = \{\pm 1\}$ , and consider a bipartite ranking problem where the goal is to learn a scoring function  $f: \mathcal{X} \rightarrow \mathbb{R}$ , with performance measured by the area under the ROC curve (AUC) (higher values are better):

$$\begin{aligned} \text{AUC}_D[f] &= \mathbf{E}[\mathbf{1}((Y - Y')(f(X) - f(X')) > 0) \\ &\quad + \frac{1}{2} \mathbf{1}(f(X) = f(X')) | Y \neq Y'], \end{aligned}$$

where  $(X, Y), (X', Y')$  are drawn i.i.d. from  $D$ . While here again, the performance measure cannot be expressed as an expectation of loss function, one can indeed compute the Bayes optimal value of the performance measure for a given distribution (e.g. see [34]); we use this as the criterion to be optimized in feature selection:

$$\begin{aligned} C_D^{\text{Bayes}, \text{AUC}}(\mathcal{J}) &= \sup_{f_{\mathcal{J}}: \mathcal{X}_{\mathcal{J}} \rightarrow \mathbb{R}} \text{AUC}_{D_{\mathcal{J}}}[f_{\mathcal{J}}] \\ &= 1 - \frac{\mathbf{E}[\min(\alpha_{\mathcal{J}}(X_{\mathcal{J}}, X'_{\mathcal{J}}), \alpha_{\mathcal{J}}(X'_{\mathcal{J}}, X_{\mathcal{J}}))]}{2p(1-p)}, \end{aligned}$$

where  $\alpha_{\mathcal{J}}(Z, Z') = \eta_{\mathcal{J}}(Z)(1 - \eta_{\mathcal{J}}(Z'))$ .

### 3.2 GREEDY ALGORITHM FOR OPTIMIZING BAYES CRITERION

As noted earlier, in practice, one does not have access to the true distribution  $D$ , and therefore must optimize an approximate version of the Bayes criterion based on the training

sample  $S = ((\mathbf{x}^1, y^1), \dots, (\mathbf{x}^m, y^m))$ . In particular, for a label space  $\mathcal{Y}$ , prediction space  $\widehat{\mathcal{Y}}$ , and loss  $\ell: \mathcal{Y} \times \widehat{\mathcal{Y}} \rightarrow \mathbb{R}_+$ , note that the Bayes  $\ell$ -error w.r.t.  $D$  can be written as

$$\text{er}_D^{\ell, *} = \mathbf{E}_X \left[ \inf_{\widehat{y} \in \widehat{\mathcal{Y}}} \mathbf{E}_{Y|X}[\ell(Y, \widehat{y})] \right].$$

To obtain a sample-based estimate of  $\text{er}_D^{\ell, *}$ , one can replace the outer expectation over  $X$  by an average over the training instances  $\mathbf{x}^i$  in  $S$ , and use an empirical estimate of the conditional distribution of  $Y$  given  $X$  in computing the inner expectation:

$$\widehat{\text{er}}_S^{\ell, *} = \frac{1}{m} \sum_{i=1}^m \inf_{\widehat{y} \in \widehat{\mathcal{Y}}} \widehat{\mathbf{E}}_{Y|X=\mathbf{x}^i}[\ell(Y, \widehat{y})],$$

where  $\widehat{\mathbf{E}}_{Y|X}$  denotes expectation with respect to an approximate conditional distribution  $\widehat{\mathbf{P}}(Y|X)$  estimated from the sample  $S$ . This gives the approximate Bayes criterion

$$\begin{aligned} \widehat{C}_S^{\text{Bayes}, \ell}(\mathcal{J}) &= -\widehat{\text{er}}_{S_{\mathcal{J}}}^{\ell, *} \\ &= -\frac{1}{m} \sum_{i=1}^m \inf_{\widehat{y} \in \widehat{\mathcal{Y}}} \widehat{\mathbf{E}}_{Y|X_{\mathcal{J}}=\mathbf{x}_{\mathcal{J}}^i}[\ell(Y, \widehat{y})], \end{aligned}$$

where again  $\widehat{\mathbf{E}}_{Y|X_{\mathcal{J}}}$  denotes expectation with respect to an approximate conditional distribution  $\widehat{\mathbf{P}}(Y|X_{\mathcal{J}})$  estimated from the sample  $S_{\mathcal{J}}$ . For example, for binary classification with 0-1 error, one gets the approximate criterion:

$$\widehat{C}_S^{\text{Bayes}, 0-1}(\mathcal{J}) = -\frac{1}{m} \sum_{i=1}^m \min(\widehat{\eta}_{\mathcal{J}}(\mathbf{x}_{\mathcal{J}}^i), 1 - \widehat{\eta}_{\mathcal{J}}(\mathbf{x}_{\mathcal{J}}^i)),$$

where  $\widehat{\eta}_{\mathcal{J}}: \mathcal{X}_{\mathcal{J}} \rightarrow [0, 1]$  is a suitable estimate of  $\eta_{\mathcal{J}}$  based on  $S_{\mathcal{J}}$ . An ideal algorithm would then select the best subset of  $k$  features according to the above approximate criterion:

$$\widehat{\mathcal{J}}_S \in \operatorname{argmax}_{\substack{\mathcal{J} \subseteq [n] \\ |\mathcal{J}|=k}} \widehat{C}_S^{\text{Bayes}, \ell}(\mathcal{J}).$$

However, this optimization problem (over  $\binom{n}{k}$  subsets) is typically still hard due to its combinatorial nature. As is often done in other feature selection approaches, one possibility is to use an algorithm that selects features to maximize the above criterion in a greedy fashion. For example, one can use a forward selection algorithm which starts with an empty feature set, and at each iteration, adds the feature with the highest marginal value of the objective  $\widehat{C}_S^{\text{Bayes}, \ell}$  to the current set of features (see Algorithm 1).<sup>3</sup>

**Conditional probability estimation for large  $k$  using  $s$ -variate approximations.** Applying the above algorithm

<sup>3</sup>We note that the proposed greedy method easily extends to settings where the value of  $k$  is not available to us; for example, one can terminate this method based on an appropriate stopping criterion (such as when the difference in feature criterion across two successive iterations falls below a certain value) and use the features chosen up to that point to learn a suitable predictor.

Table 1: Data sets used in our experiments.

| Data set | No. of features | No. of instances | Feature type | $p = \mathbf{P}(Y = 1)$ |
|----------|-----------------|------------------|--------------|-------------------------|
| Mushroom | 116             | 8124             | Binary       | 0.482                   |
| Adult    | 123             | 48824            | Binary       | 0.239                   |
| Splice   | 240             | 3190             | Binary       | 0.519                   |
| Semeion  | 256             | 1593             | Binary       | 0.102                   |
| KDDCup01 | 139351          | 1909             | Binary       | 0.022                   |
| Pcmac    | 3289            | 1943             | Integer      | 0.495                   |
| Basehock | 4862            | 1993             | Integer      | 0.501                   |
| Gisette  | 5000            | 6000             | Integer      | 0.500                   |
| Waveform | 40              | 5000             | Real         | 0.331                   |

as shown requires computing conditional probability estimates  $\hat{\mathbf{P}}(Y|X_{\mathcal{J}})$  for feature sets  $\mathcal{J}$  of size up to  $k$ . For small  $k$ , this is easy to do; for example, for problems with binary labels, one computes:

$$\hat{\eta}_{\mathcal{J}}(\mathbf{z}) = \hat{\mathbf{P}}(Y = 1 | X_{\mathcal{J}} = \mathbf{z}) = \begin{cases} \frac{\sum_{i=1}^m \mathbf{1}(\mathbf{x}_{\mathcal{J}}^i = \mathbf{z}, y^i = 1)}{\sum_{i=1}^m \mathbf{1}(\mathbf{x}_{\mathcal{J}}^i = \mathbf{z})} & \text{if } \sum_{i=1}^m \mathbf{1}(\mathbf{x}_{\mathcal{J}}^i = \mathbf{z}) > 0 \\ \frac{1}{2} & \text{otherwise.} \end{cases}$$

When  $k$  is large, one runs into difficulties in later iterations of the algorithm. Specifically, consider the  $t$ -th iteration, when  $(t - 1) < k$  features  $j_1, \dots, j_{t-1}$  have been added to  $\mathcal{J}$  and the  $t$ -th feature is to be selected. For large  $t$ , it is likely that most configurations of  $\mathbf{x}_{\mathcal{J}}^i$  appear only once in the training sample, and therefore for all potential features  $j_t \in [n] \setminus \mathcal{J}$ , one gets (in a setting with binary labels) that  $\hat{\eta}_{\mathcal{J} \cup \{j_t\}}(\mathbf{x}_{\mathcal{J} \cup \{j_t\}}^i)$  is either 0, 1 or  $\frac{1}{2}$ , thus giving many ties and no useful basis for selecting the next feature. This is an inherent difficulty that arises when estimating high-dimensional multivariate distributions from limited data. A common approach to overcome this problem, often used in the context of optimizing the MI criterion (e.g. see [5]), is to use approximate calculations that require estimating conditional distributions on only smaller subsets of the features; one such approach is a  $s$ -variate approximation (for some small  $s < k$ ), where the given filter criterion on a set of  $k$  features  $\mathcal{J}$  is approximated by the average value of the criterion on all subsets of  $\mathcal{J}$  of size  $s$  [20]:

$$\hat{C}_S^{\text{Bayes}}(\mathcal{J}) \approx \frac{1}{\binom{k}{s}} \sum_{A \subset \mathcal{J}, |A|=s} \hat{C}_S^{\text{Bayes}}(A).$$

With such approximations, one can use algorithms based on both forward selection and backward elimination to greedily maximize the Bayes criterion. In our experiments with large feature subsets, we use the standard bivariate approximation with  $s = 2$ .

## 4 EXPERIMENTS

We now report results of experiments designed to evaluate the proposed Bayes optimal feature selection method in

a variety of settings with different performance measures. These include binary classification with both the standard 0-1 and cost-sensitive losses, binary class probability estimation (CPE) with the logarithmic loss (under which our Bayes criterion reduces to MI criterion), and learning under class imbalance with the balanced 0-1 loss and F-measure. The data sets used in our experiments are shown in Table 1; these include varying numbers of features/examples, feature types, and class probabilities.<sup>4</sup> Each data set was split into train-test sets, with the feature selection methods and learning algorithms applied on the training set, and the learned model evaluated on the test set; the average performance over 5 random train-test splits is then reported. All tunable parameters in the learning algorithms used were chosen using a held-out portion of the training set.<sup>5,6</sup>

**Baselines.** Our main method, which optimizes the Bayes criterion corresponding to the loss or performance measure of interest in a greedy manner (possibly with some approximations in estimating high-dimensional conditional distributions), is termed BayesGreedy. We also include a score-based variant of our method (BayesScore) that scores each feature independently using the Bayes criterion evaluated on the corresponding one-dimensional feature space, and selects the top  $k$  features according to this score. As baselines, we consider a number of standard filter methods popular in practice. These include a method that optimizes the MI criterion in a greedy manner (again with some approximations in estimating high-dimensional conditional distributions), termed MIGreedy [5, 7]; a score-based vari-

<sup>4</sup>We obtained Pcmac and Basehock from the ASU repository (<http://featureselection.asu.edu>), KDDCup01 from the KDD Cup Challenge 2001 (<http://pages.cs.wisc.edu/~kddcup2001/>) and the rest from the UCI ML repository (<http://www.ics.uci.edu/~mllearn/MLRepository.html>). Of these, Semeion and Waveform are multi-class data sets, where one of the class was taken as positive, and the remaining were combined into the negative class.

<sup>5</sup>In the case of the larger Adult data set, 20% of the data was used for training and the remaining for testing. On all other data sets, 70% was used for training. In each case, a held-out 20% of the training set was used for parameter tuning.

<sup>6</sup>For data sets with integer/real valued features, we discretized each feature into three categories based on intervals:  $(-\infty, \mu - \sigma)$ ,  $[\mu - \sigma, \mu + \sigma)$ , and  $[\mu + \sigma, \infty)$ , where  $\mu$  is the mean feature value and  $\sigma$  is the standard deviation.

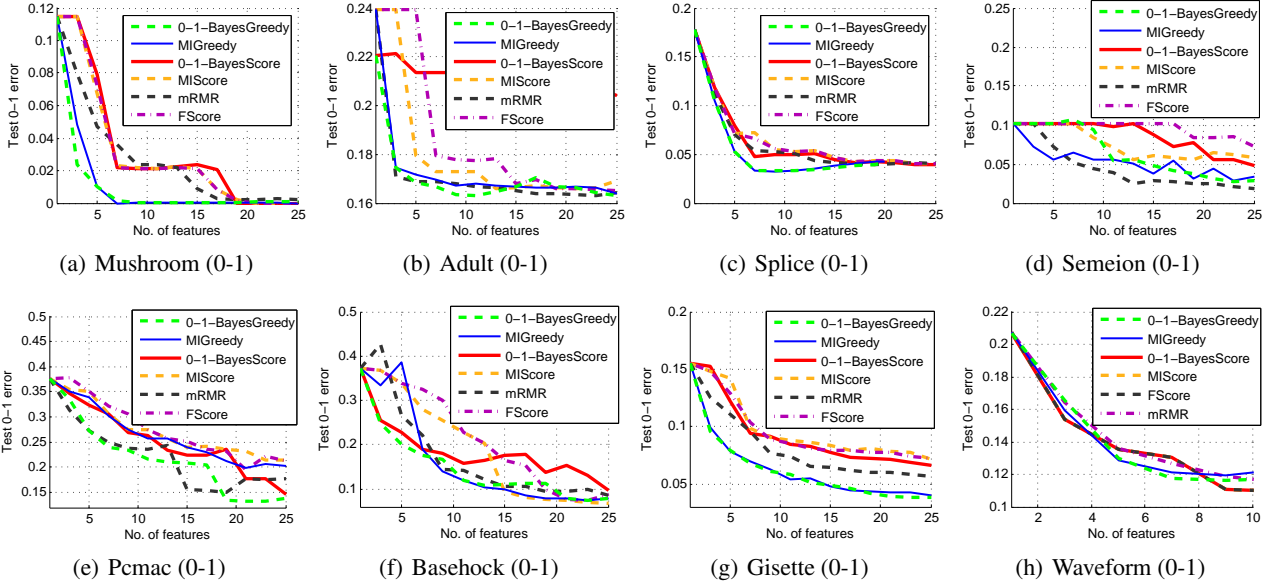


Figure 1: Feature selection for binary 0-1 classification. Plots show test 0-1 error vs. number of features for different feature selection methods, with SVM (RBF kernel) as the classification algorithm.

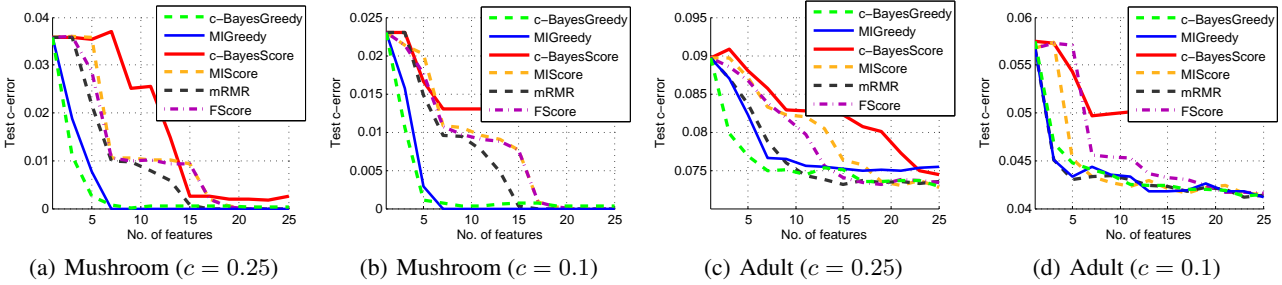


Figure 2: Feature selection for cost-sensitive binary classification with different costs  $c$ . Plots show test cost-sensitive error vs. number of features for various filter methods, with cost-sensitive SVM (RBF kernel) as the classification algorithm.

ant of this method for optimizing the MI criterion (MIScore) [21]; and a score-based method that optimizes another popular feature selection criterion, namely the Fischer score (F-score) [35]. Apart from the above methods, there are other filter methods based on MI that in addition to optimizing for relevant feature subsets, also seek to promote some form of ‘diversity’ among the chosen features. Popular among these is the minimal-redundancy-maximal-relevance (mRMR) method [6], which we include as a representative baseline from this category.

The above baselines are indeed representative of the various filter methods used in practice, with most other methods based on MI being variants of the MIGreedy or mRMR methods. Since the focus of this paper is entirely on filter methods, we do not compare our approach against wrapper or embedded methods, which unlike filter methods are closely tied to the learning algorithm used.

In experiments below, unless otherwise specified, the BayesGreedy and MIGreedy methods shall use exact estimates of class-conditional distributions.

#### 4.1 BINARY CLASSIFICATION (0-1 ERROR AND COST-SENSITIVE ERROR)

The first task that we consider is binary classification with the standard 0-1 error (see Example 2 for the Bayes criterion for this performance measure). We used kernel SVM (with RBF kernel) as the learning algorithm for this task. Figure 1 contains the test 0-1-error for the different feature selection methods as a function of the number of features chosen. As seen, on all data sets except Semeion and for most feature subset sizes, the features chosen by the proposed BayesGreedy method (that explicitly optimizes the 0-1 error) perform comparable to or better than the baseline methods. The poor performance of BayesGreedy on the Semeion data set was due to the inexact/greedy search technique used by the method (when the Bayes criterion was optimized exactly on this data set using an exhaustive search over feature subsets, we did obtain better performance than the MI criterion).

We also consider the task of binary classification with cost-sensitive error (see Example 3 for the Bayes criterion).

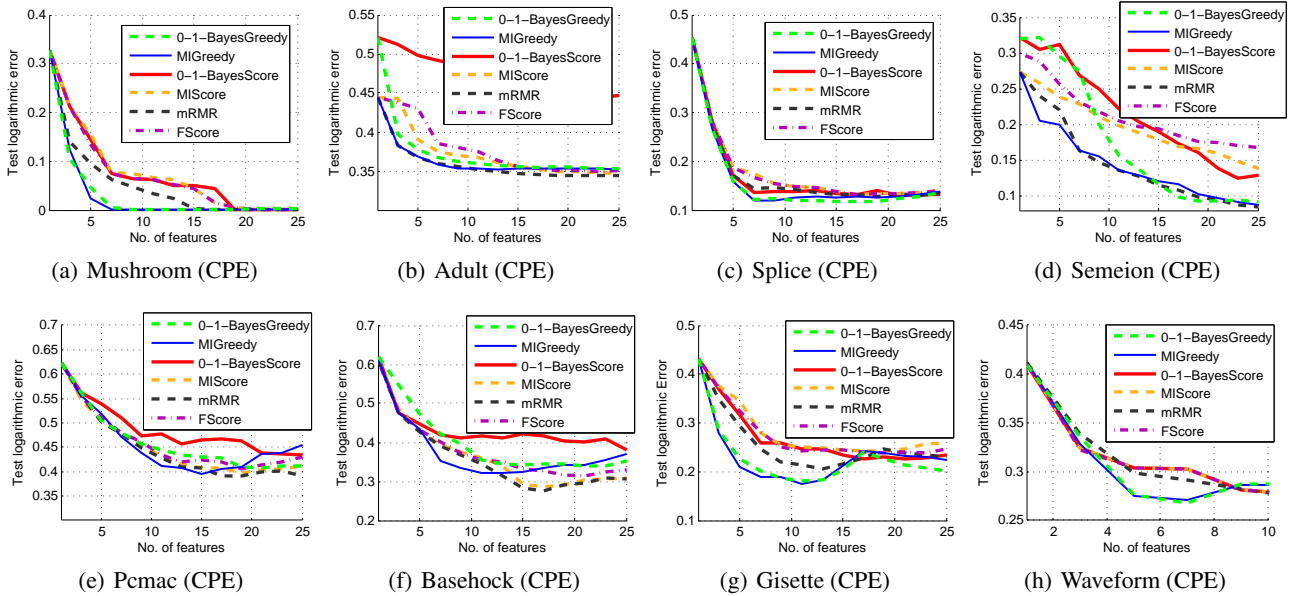


Figure 3: Feature selection for binary CPE. Plots show test logarithmic error vs. number of features for different feature selection methods, with logistic regression (RBF kernel) as the CPE algorithm. Here, MI is the Bayes optimal criterion for the logarithmic error; one can see that MIGreedy performs the best in most cases.

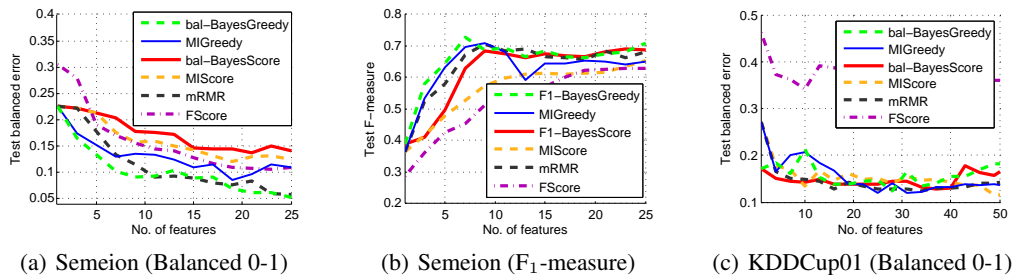


Figure 4: Feature selection for learning under class imbalance. (a), (c) Test balanced 0-1 error vs. number of features, with balanced SVM (RBF kernel) used as the learning algorithm. (b) Test  $F_1$ -measure vs. number of features, with a plug-in method that uses logistic regression (RBF kernel) followed by empirical thresholding as the learning algorithm. Bivariate approximations were used in estimating class-conditionals for KDDCup01. Higher values are better for  $F_1$ -measure.

We used cost-sensitive kernel SVM as the learning algorithm here. Figure 2 contains results on the Adult and Mushroom data sets with different costs. In three of four cases, BayesGreedy yields lower cost-sensitive error than the baselines for smaller feature subset sizes and comparable values for larger feature subset sizes.

On most data sets, the score-based methods do not perform as well as the other methods; this is due to their naive search strategy where the features are scored independently.

#### 4.2 BINARY CLASS PROBABILITY ESTIMATION (LOGARITHMIC ERROR)

The next task that we consider is class probability estimation with the logarithmic error. As mentioned earlier, the Bayes criterion here effectively reduces to the MI criterion (see Example 5). We used regularized kernel logistic regression (with RBF kernel) as the class probability estima-

tion algorithm here. Figure 3 contains plots of the test logarithmic error vs. the number of features chosen for different feature selection methods; we also include for comparison methods that optimize the Bayes criterion for the 0-1 loss. MIGreedy, which optimizes the Bayes criterion for the logarithmic error, performs comparable to or better than the other methods for most feature subset sizes.

#### 4.3 LEARNING UNDER CLASS IMBALANCE (BALANCED 0-1 ERROR AND F-MEASURE)

We now move to the task of binary classification under class imbalance. Commonly used performance measures in this setting include the balanced 0-1-error and  $F_1$ -measure, both of which aim to balance errors on either classes (see Examples 4 and 7 for the Bayes criterion for these measures). We used a balanced version of SVM (where the positive and negative points were weighted with costs  $1/p$  and  $1/(1-p)$  respectively) as the learning algorithm for

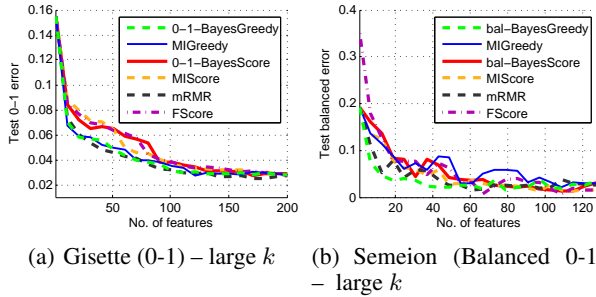


Figure 5: Selecting larger numbers of features. The settings here are similar to previous plots, except that bivariate approximations were used in estimating class-conditionals.

the balanced 0-1-error; and a plug-in method using logistic regression followed by thresholding of the resulting class probability estimate at a sample-based optimal point as the learning algorithm for the  $F_1$ -measure [33, 36, 37]. Figure 4 contains results on the class-imbalanced Semeion ( $p = 0.102$ ) and KDDCup01 ( $p = 0.022$ ) data sets. In the case of Semeion, the BayesGreedy methods perform the best over all. With KDDCup01, where we include results for the balanced 0-1 error (the performance measure used in the KDD Cup 2001 challenge), there is no clear winner; here, BayesScore performs the best for smaller feature subsets, and MIGreedy performs better for larger subsets.

#### 4.4 SELECTING LARGER NUMBER OF FEATURES

We also evaluated the proposed filter methods on large feature subset sizes  $k$ . As noted earlier, an exact implementation of the prescribed greedy algorithm is difficult in this case as estimation of high-dimensional class-conditional distributions from limited data is prone to errors and is also computationally expensive. We therefore resorted to the bivariate approximation technique described in Section 3.2 for estimating the class-conditional distributions; the MIGreedy method also used the same estimation procedure, while the search technique in mRMR inherently used a similar approximation [6]. Figure 5 contains results on the Gisetite (binary classification with 0-1 loss) and Semeion (binary classification with balanced 0-1 loss) data sets. In the case of Semeion, the BayesGreedy method consistently performs as well as (if not better than) the baselines; in the case of Gisetite, BayesGreedy is the second best over all.

#### 4.5 RUN-TIME COMPARISONS

We now present run-time comparisons of the various filter methods for different values of  $k$ . Table 2 contains the run-times (in seconds) for cost-sensitive classification on Adult data, and 0-1 binary classification with large  $k$  on Gisetite data (with approximations used to estimate conditional distributions). All methods here were implemented in MATLAB. As expected, the score-based methods, requiring only a single sort operation, offer the least run-

| Adult ( $c = 0.25$ )       |          |          |          |
|----------------------------|----------|----------|----------|
|                            | $k = 5$  | $k = 10$ | $k = 15$ |
| $c$ -BayesGreedy           | 1.75     | 12.78    | 71.14    |
| MIGreedy                   | 1.95     | 15.63    | 94.06    |
| $c$ -BayesScore            | 0.10     | 0.10     | 0.10     |
| MIScore                    | 0.17     | 0.19     | 0.17     |
| mRMR                       | 1.32     | 5.27     | 11.71    |
| FScore                     | 0.06     | 0.08     | 0.08     |
| Gisetite (0-1) – large $k$ |          |          |          |
|                            | $k = 25$ | $k = 50$ | $k = 75$ |
| 0-1-BayesGreedy            | 954      | 3784     | 8475     |
| MIGreedy                   | 2123     | 8330     | 18475    |
| 0-1-BayesScore             | 2.62     | 2.65     | 2.71     |
| MIScore                    | 2.67     | 2.73     | 2.76     |
| mRMR                       | 1242     | 5046     | 11384    |
| FScore                     | 1.01     | 1.08     | 1.04     |

Table 2: Run-time comparison of various filter methods for different values of  $k$ . All values are in *seconds*. The settings here are same as before. For Gisetite, bivariate approximations were used to estimate conditional distributions.

times; however, as seen earlier, these methods often perform poorly in terms of accuracy. Among the other methods, BayesGreedy is significantly faster than MIGreedy, despite both methods using the same search procedure (this is because the Bayes criteria for the 0-1 and cost-sensitive losses involve simple ‘max’ operations that can be implemented efficiently). On the Adult data, where BayesGreedy computes exact estimates of conditional distributions, it is slower than mRMR; however, when BayesGreedy uses computationally cheaper bivariate approximations to estimate probabilities, it yields lower run-times than mRMR even for larger values of  $k$ , as seen with the Gisetite data.

## 5 CONCLUSION

We have developed a Bayes optimal filter method for feature selection with supervised learning considering general performance measures, and provided instantiations of our method for a variety of learning problems and performance measures. Experiments demonstrate that our approach is competitive with many state-of-the-art methods. While our focus has been on problems with binary labels, our approach easily generalizes to multiclass settings.

A possible direction of work in the future is to investigate approximation guarantees for the greedy algorithm used to optimize a given Bayes optimal criteria. Indeed (under specific assumptions) such guarantees have been established for the MI criterion and the criterion for regression with squared loss, by leveraging tools from submodular optimization [38, 39]. It would be interesting to explore similar results for the other filter criteria developed in this work.

**Acknowledgements.** HN acknowledges support from a Google India PhD fellowship. SA thanks DST for support under a Ramanujan Fellowship.

## References

- [1] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.
- [2] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3):389–422, 2002.
- [3] Y. Yang and J.O. Pedersen. A comparative study on feature selection in text categorization. In *ICML*, 1997.
- [4] R. Battiti. Using mutual information for selecting features in supervised neural net learning. *IEEE Transactions on Neural Networks*, 5(4):537–550, 1994.
- [5] D. Koller and M. Sahami. Toward optimal feature selection. In *ICML*, 1996.
- [6] H. Peng, F. Long, and C. Ding. Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:1226–1238, 2005.
- [7] G. Brown, A. Pocock, M-J. Zhao, and M. Luján. Conditional likelihood maximisation: A unifying framework for information theoretic feature selection. *Journal of Machine Learning Research*, 13:27–66, 2012.
- [8] W. Duch. Filter methods. In Isabelle Guyon, Steve Gunn, Masoud Nikravesh, and Lofti Zadeh, editors, *Feature Extraction: Foundations and Applications*. Springer, 2006.
- [9] G. Saon and M. Padmanabhan. Minimum Bayes error feature selection for continuous speech recognition. In *NIPS*, 2000.
- [10] L.C. Molina, L. Belanche, and À. Nebot. Feature selection algorithms: A survey and experimental evaluation. In *ICDM*, 2002.
- [11] G. Carneiro and N. Vasconcelos. Minimum Bayes error features for visual recognition by sequential feature selection and extraction. In *CRV*, 2005.
- [12] S-H. Yang, H. Zha, S.K. Zhou, and B-G. Hu. Variational graph embedding for globally and locally consistent feature extraction. In *ECML PKDD*. 2009.
- [13] S-H. Yang and B-G. Hu. Discriminative feature selection by nonparametric Bayes error minimization. *IEEE Transactions on Knowledge and Data Engineering*, 24(8):1422–1434, 2012.
- [14] M. Robnik-Šikonja and I. Kononenko. Theoretical and empirical analysis of relief and rrelief. *Machine learning*, 53(1-2):23–69, 2003.
- [15] Q. Gu, Z. Li, and J. Han. Generalized fisher score for feature selection. In *UAI*, 2011.
- [16] D.A. Bell and H. Wang. A formalism for relevance and its application in feature subset selection. *Machine learning*, 41(2):175–195, 2000.
- [17] N. Kwak and C-H. Choi. Input feature selection for classification problems. *IEEE Transactions on Neural Networks*, 13(1):143–159, 2002.
- [18] F. Fleuret. Fast binary feature selection with conditional mutual information. *Journal of Machine Learning Research*, 5:1531–1555, 2004.
- [19] L. Yu and H. Liu. Efficient feature selection via analysis of relevance and redundancy. *Journal of Machine Learning Research*, 5:1205–1224, 2004.
- [20] P.E. Meyer, C. Schretter, and G. Bontempi. Information-theoretic feature selection in microarray data using variable complementarity. *Journal of Selected Topics in Signal Processing*, 2(3):261–274, 2008.
- [21] N. Vasconcelos. Feature selection by maximum marginal diversity. In *NIPS*, 2002.
- [22] R.M. Fano. *Transmission of information: A statistical theory of communications*. M.I.T. Press, 1961.
- [23] L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer, 1996.
- [24] M-J. Zhao, N. Edakunni, A. Pocock, and G. Brown. Beyond Fano’s inequality: Bounds on the optimal F-score, BER, and cost-sensitive risk and their implications. *Journal of Machine Learning Research*, 14(1):1033–1090, 2013.
- [25] G. Forman. An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research*, 3:1289–1305, 2003.
- [26] Z. Zheng, X. Wu, and R. Srihari. Feature selection for text categorization on imbalanced data. *ACM SIGKDD Explorations Newsletter*, 6(1):80–89, 2004.
- [27] X. Geng, T-Y. Liu, T. Qin, and H. Li. Feature selection for ranking. In *SIGIR*, 2007.
- [28] R. Kohavi and G.H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1):273–324, 1997.
- [29] I. Tsamardinos and C.F. Aliferis. Towards principled feature selection: Relevancy, filters and wrappers. In *AISTATS*, 2003.
- [30] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.
- [31] A.K. Menon, H. Narasimhan, S. Agarwal, and S. Chawla. On the Statistical Consistency of Algorithms for Binary Classification under Class Imbalance. In *ICML*, 2013.
- [32] A. Miller. *Subset Selection in Regression*. Chapman and Hall, 2002.
- [33] N. Ye, K.M.A. Chai, W.S. Lee, and H.L. Chieu. Optimizing F-measures: A tale of two approaches. In *ICML*, 2012.
- [34] S. Cléménçon, G. Lugosi, and N. Vayatis. Ranking and empirical minimization of U-statistics. *Annals of Statistics*, 36:844–874, 2008.
- [35] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. Wiley-Interscience, 2000.
- [36] H. Narasimhan, R. Vaish, and S. Agarwal. On the statistical consistency of plug-in classifiers for non-decomposable performance measures. In *NIPS*, 2014.
- [37] O. Koyejo, N. Natarajan, P. Ravikumar, and I.S. Dhillon. Consistent binary classification with generalized performance metrics. In *NIPS*, 2014.
- [38] A. Krause and C.E. Guestrin. Near-optimal nonmyopic value of information in graphical models. 2005.
- [39] A. Das and D. Kempe. Submodular meets spectral: Greedy algorithms for subset selection, sparse approximation and dictionary selection. In *ICML*, 2011.

---

# Visual Causal Feature Learning

---

**Krzysztof Chalupka**  
Computation and Neural Systems  
California Institute of Technology  
Pasadena, CA, USA

**Pietro Perona**  
Electrical Engineering  
California Institute of Technology  
Pasadena, CA, USA

**Frederick Eberhardt**  
Humanities and Social Sciences  
California Institute of Technology  
Pasadena, CA, USA

## Abstract

We provide a rigorous definition of the *visual cause* of a behavior that is broadly applicable to the visually driven behavior in humans, animals, neurons, robots and other perceiving systems. Our framework generalizes standard accounts of causal learning to settings in which the causal variables need to be constructed from micro-variables. We prove the Causal Coarsening Theorem, which allows us to gain causal knowledge from observational data with minimal experimental effort. The theorem provides a connection to standard inference techniques in machine learning that identify features of an image that *correlate* with, but may not *cause*, the target behavior. Finally, we propose an active learning scheme to learn a manipulator function that performs optimal manipulations on the image to automatically identify the visual cause of a target behavior. We illustrate our inference and learning algorithms in experiments based on both synthetic and real data.

## 1 INTRODUCTION

Visual perception is an important trigger of human and animal behavior. The visual cause of a behavior can be easy to define, say, when a traffic light turns green, or quite subtle: apparently it is the increased symmetry of features that leads people to judge faces more attractive than others (Grammer and Thornhill, 1994). Significant scientific and economic effort is focused on visual causes in advertising, entertainment, communication, design, medicine, robotics and the study of human and animal cognition. Visual causes profoundly influence our daily activity, yet our understanding of what constitutes a visual cause lacks a theoretical basis. In practice, it is well-known that images are composed of millions of variables (the pixels) but it is functions of the pixels (often called ‘features’) that have meaning, rather than the pixels themselves.

We present a theoretical framework and inference algorithms for visual causes in images. A visual cause is defined (more formally below) as a function (or *feature*) of raw image pixels that has a *causal effect* on the target behavior of a perceiving system of interest. We present three advances:

- We provide a definition of the visual cause of a target behavior as a macro-variable that is constructed from the micro-variables (pixels) that make up the image space. The visual cause is distinguished from other macro-variables in that it contains all the causal information about the target behavior that is available in the image. We place the visual cause within the standard framework of causal graphical models (Spirtes et al., 2000; Pearl, 2009), thereby contributing to an account of how to construct causal variables.
- We prove the Causal Coarsening Theorem (CCT), which shows how observational data can be used to learn the visual cause with minimal experimental effort. It connects the present results to standard classification tasks in machine learning.
- We describe a method to learn the manipulator function, which automatically performs perceptually optimal manipulations on the visual causes.

We illustrate our ideas using synthetic and real-data experiments. Python code that implements our algorithms, as well as reproduces some of the experimental results, is available online at <http://vision.caltech.edu/~kchalupk/code.html>.

We chose to develop the theory within the context of *visual* causes as this setting makes the definitions most intuitive and is itself of significant practical interest. However, the framework and results can be equally well applied to extract causal information from any aggregate of micro-variables on which manipulations are possible. Examples include auditory, olfactory and other sensory stimuli; high-dimensional neural recordings; market data in finance; consumer data in marketing. There, causal feature learning is both of theoretical (“What is the cause?”) and practical (“Can we automatically manipulate it?”) importance.

## 1.1 PREVIOUS WORK

Our framework extends the theory of causal graphical models (Spirtes et al., 2000; Pearl, 2009) to a setting in which the input data consists of raw pixel (or other micro-variable) data. In contrast to the standard setting, in which the macro-variables in the statistical dataset already specify the candidate causal relations, the causal variables in our setting have to be constructed from the micro-variables they supervene on, before any causal relations can be established. We emphasize the difference between our method of causal feature *learning* and methods for causal feature *selection* (Guyon et al., 2007; Pellet and Elisseeff, 2008). The latter choose the best (under some causal criterion) features from a restricted set of plausible macro-variable candidates. In contrast, our framework efficiently searches the whole space of all the possible macro-variables that can be constructed from an image.

Our approach derives its theoretical underpinnings from the theory of computational mechanics (Shalizi and Crutchfield, 2001; Shalizi, 2001), but supports a more explicitly causal interpretation by incorporating the possibility of confounding and interventions. We take the distinction between interventional and observational distributions to be one of the key features of a causal analysis. Since we allow for unmeasured common causes of the features in the image and the target behavior, we have to distinguish between the plain conditional probability distribution of the target behavior ( $T$ ) given the (observed) image ( $I$ ) and the distribution of the target behavior given that the observed image was manipulated (i.e.  $P(T|I)$  vs.  $P(T|do(I))$ ). Hoel et al. (2013), who develop a similar model to investigate the relationship between causal micro- and macro-variables, avoid this distinction by assuming that all their data was generated from what in our setting would be the manipulated distribution  $P(T|do(I))$ . The extant literature on causal learning from image or video data does not generally consider the aggregation from pixel variables into causal macro-variables, but instead starts from annotated or pre-defined features of the image (see e.g. Fire and Zhu (2013a,b)).

## 1.2 CAUSAL FEATURE LEARNING: AN EXAMPLE

Fig. 1 presents a paradigmatic case study in visual causal feature learning, which we will use as a running example. The contents of an image  $I$  are caused by external, non-visual binary hidden variables  $H_1$  and  $H_2$  such that if  $H_1$  is on,  $I$  contains a vertical bar (v-bar<sup>1</sup>) at a random position, and if  $H_2$  is on,  $I$  contains a horizontal bar (h-bar) at a random position. A target behavior  $T \in \{0, 1\}$  is caused by  $H_1$  and  $I$ , such that  $T = 1$  is more likely whenever  $H_1 = 1$  and whenever the image contains an h-bar.

<sup>1</sup>We take a v-bar (h-bar) to consist of a complete column (row) of black pixels.

We deliberately constructed this example such that the visual cause is clearly identifiable: manipulating the presence of an h-bar in the image will influence the distribution of  $T$ . Thus, we can call the following function  $C: \mathcal{I} \rightarrow \{0, 1\}$  the *causal feature* of  $I$  or the *visual cause* of  $T$ :

$$C(I) = \begin{cases} 1 & \text{if } I \text{ contains an h-bar} \\ 0 & \text{otherwise.} \end{cases}$$

The presence of a v-bar, on the other hand, is not a causal feature. Manipulating the presence of a v-bar in the image has no effect on  $H_1$  or  $T$ . Still, the presence of a v-bar is as strongly correlated with the value of  $T$  (via the common cause  $H_1$ ) as the presence of an h-bar is. We will call the following function  $S: \mathcal{I} \rightarrow \{0, 1\}$  the *spurious correlate* of  $T$  in  $I$ :

$$S(I) = \begin{cases} 1 & \text{if } I \text{ contains a v-bar} \\ 0 & \text{otherwise.} \end{cases}$$

Both the presence of h-bars and the presence of v-bars are good individual (and even better joint) predictors of the target variable, but only one of them is a cause. Identifying the visual cause from the image thus requires the ability to distinguish among the correlates of the target variables those that are actually causal, even if the non-causal correlates are (possibly more strongly) correlated with the target.

While the values of  $S$  and  $C$  in our example stand in a bijective correspondence to the values of  $H_1$  and  $H_2$ , respectively, this is only to keep the illustration simple. In general, the visual cause and the spurious correlate can be probabilistic functions of any number of (not necessarily independent) hidden variables, and can share the same hidden causes.

## 2 A THEORY OF VISUAL CAUSAL FEATURES

In our example the identification of the visual cause with the presence of an h-bar is intuitively obvious, as the model is constructed to have an easily describable visual cause. But the example does not provide a theoretical account of what it takes to be a visual cause in the general case when we do not know what the causally relevant pixel configurations are. In this section, we provide a general account of how the visual cause is related to the pixel data.

### 2.1 VISUAL CAUSES AS MACRO-VARIABLES

A visual cause is a high-level random variable that is a function (or feature) of the image, which in turn is defined by the random micro-variables that determine the pixel values. The functional relation between the image and the visual cause is, in general, surjective, though in principle it could be bijective. While we are interested in identifying



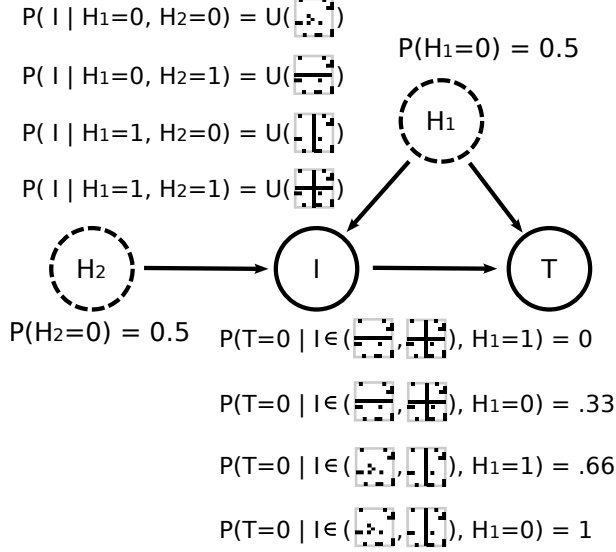


Figure 1: Our case study generative model. Two binary hidden (non-visual) variables  $H_1$  and  $H_2$  toss unbiased coins. The content of the image  $I$  depends on these variables as follows. If  $H_1 = H_2 = 0$ ,  $I$  is chosen uniformly at random from all the images containing no v-bars and no h-bars. If  $H_1 = 0$  and  $H_2 = 1$ ,  $I$  is chosen uniformly at random from all images containing at least one h-bar but no v-bars. If  $H_1 = 1$  and  $H_2 = 0$ ,  $I$  is chosen uniformly at random from all the images containing at least one v-bar but no h-bars. Finally, if  $H_1 = H_2 = 1$ ,  $I$  is chosen from images containing at least one v-bar and at least one h-bar. The distribution of the binary behavior  $T$  depends only on the presence of an h-bar in  $I$  and the value of  $H_1$ . In observational studies,  $H_1 = 1$  iff  $I$  contains a v-bar. However, a *manipulation* of any specific image  $I = i$  that introduces a v-bar (without changing  $H_1$ ) will in general not change the probability of  $T$  occurring. Thus,  $T$  does *not* depend causally on the presence of v-bars in  $I$ .

the visual causes of a target behavior, the functional relation between the image pixels and the visual cause should not itself be interpreted as causal. Pixels do not *cause* the features of an image, they *constitute* them, just as the atoms of a table constitute the table (and its features). The difference between the causal and the constitutive relation is that the former requires the possibility of independent manipulation (at least to some extent), whereas by definition one cannot manipulate the visual cause without manipulating the image pixels.

The probability distribution over the visual cause is induced by the probability distribution over the pixels in the image and the functional mapping from the image to the visual cause. But since a visual cause stands in a constitutive relation with the image, we cannot without further explanation describe interventions on the visual cause in terms of the standard *do*-operation (Pearl, 2009). Our goal will be to define a macro-variable  $C$ , which contains all the causal

information available in an image about a given behavior  $T$ , and define its manipulation. To make the problem approachable, we introduce two (natural) assumptions about the causal relation between the image and the behavior: (i) The value of the target behavior  $T$  is determined subsequently to the image in time, and (ii) the variable  $T$  is in no way represented in the image. These assumptions exclude the possibility that  $T$  is a cause of features in the image or that  $T$  can be seen as causing itself.

## 2.2 GENERATIVE MODELS: FROM MICRO- TO MACRO-VARIABLES

Let  $T \in \{0, 1\}$  represent a target behavior.<sup>2</sup> Let  $\mathcal{I}$  be a discrete space of all the images that can influence the target behavior (in our experiments in Section 4,  $\mathcal{I}$  is the space of  $n$ -dimensional black-and-white images). We use the following generative model to describe the relation between the images and the target behavior: An image is generated by a finite set of unobserved discrete variables  $H_1, \dots, H_m$  (we write  $\mathbf{H}$  for short). The target behavior is then determined by the image and possibly a subset of variables  $\mathbf{H}_c \subseteq \mathbf{H}$  that are confounders of the image and the target behavior:

$$\begin{aligned}
 P(T, I) &= \sum_{\mathbf{H}} P(T | I, \mathbf{H}) P(I | \mathbf{H}) P(\mathbf{H}) \\
 &= \sum_{\mathbf{H}} P(T | I, \mathbf{H}_c) P(I | \mathbf{H}) P(\mathbf{H}). \quad (1)
 \end{aligned}$$

Independent noise that may contribute to the target behavior is marginalized and omitted for the sake of simplicity in the above equation. The noise term incorporates any hidden variables which influence the behavior but stand in no causal relation to the image. Such variables are not directly relevant to the problem. Fig. 2 shows this generative model.

Under this model, we can define an *observational partition* of the space of images  $\mathcal{I}$  that groups images into classes that have the same conditional probability  $P(T | I)$ :

**Definition 1** (Observational Partition, Observational Class). *The observational partition  $\Pi_o(T, \mathcal{I})$  of the set  $\mathcal{I}$  w.r.t. behavior  $T$  is the partition induced by the equivalence relation  $\sim$  such that  $i \sim j$  if and only if  $P(T | I = i) = P(T | I = j)$ . We will denote it as  $\Pi_o$  when the context is clear. A cell of an observational partition is called an observational class.*

In standard classification tasks in machine learning, the observational partition is associated with class labels. In our case, two images that belong to the same cell of the observational partition assign equal *predictive* probability to the target behavior. Thus, knowing the observational class

<sup>2</sup>An extension of the framework to non-binary, discrete  $T$  is easy but complicates the notation significantly. An extension to the continuous case is beyond the scope of this article.

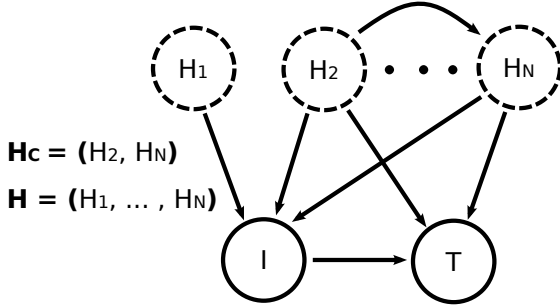


Figure 2: A general model of visual causation. In our model each image  $I$  is caused by a number of hidden non-visual variables  $H_i$ , which need not be independent. The image itself is the only observed cause of a target behavior  $T$ . In addition, a (not necessarily proper) subset of the hidden variables can be a cause of the target behavior. These confounders create visual “spurious correlates” of the behavior in  $I$ .

of an image allows us to predict the value of  $T$ . However, the predictive probability assigned to an image does not tell us the *causal* effect of the image on  $T$ . For example, a barometer is widely taken to be an excellent predictor of the weather. But changing the barometer needle does not cause an improvement of the weather. It is not a (visual or otherwise) cause of the weather. In contrast, seeing a particular barometer reading may well be a *visual cause* of whether we pack an umbrella.

Our notion of a visual cause depends on the ability to manipulate the image.

**Definition 2** (Visual Manipulation). A visual manipulation is the operation  $man(I = i)$  that changes (the pixels of) the image to image  $i \in \mathcal{I}$ , while not affecting any other variables (such as  $\mathbf{H}$  or  $T$ ). That is, the manipulated probability distribution of the generative model in Eq. (1) is given by  $P(T | man(I = i)) = \sum_{\mathbf{H}_c} P(T | I = i, \mathbf{H}_c)P(\mathbf{H}_c)$ .

The manipulation changes the values of the image pixels, but does not change the underlying “world”, represented in our model by the  $H_i$  that generated the image. Formally, the manipulation is similar to the *do*-operator for standard causal models. However, we here reserve the *do*-operation for interventions on causal *macro*-variables, such as the visual cause of  $T$ . We discuss the distinction in more detail below.

We can now define the *causal partition* of the image space (with respect to the target behavior  $T$ ) as:

**Definition 3** (Causal Partition, Causal Class). The causal partition  $\Pi_c(T, \mathcal{I})$  of the set  $\mathcal{I}$  w.r.t. behavior  $T$  is the partition induced by the equivalence relation  $\sim$  defined on  $\mathcal{I}$  such that  $i \sim j$  if and only if  $P(T | man(I = i)) = P(T | man(I = j))$  for  $i, j \in \mathcal{I}$ . When the image space and the target behavior are clear from the context, we will indicate the causal partition by  $\Pi_c$ . A cell of a causal partition is

called a causal class.

The underlying idea is that images are considered causally equivalent with respect to  $T$  if they have the same causal effect on  $T$ . Given the causal partition of the image space, we can now define the visual cause of  $T$ :

**Definition 4** (Visual Cause). The visual cause  $C$  of a target behavior  $T$  is a random variable whose value stands in a bijective relation to the causal class of  $I$ .

The visual cause is thus a function over  $\mathcal{I}$ , whose values correspond to the post-manipulation distributions  $C(i) = P(T | man(I = i))$ . We will write  $C(i) = c$  to indicate that the causal class of image  $i \in \mathcal{I}$  is  $c$ , or in other words, that in image  $i$ , the visual cause  $C$  takes value  $c$ . Knowing  $C$  allows us to predict the effects of a visual manipulation  $P(T | man(I = i))$ , as long as we have estimated  $P(T | man(I = i_k^*))$  for one representative  $i_k^*$  of each causal class  $k$ .

### 2.3 THE CAUSAL COARSENING THEOREM

Our main theorem relates the causal and observational partitions for a given  $\mathcal{I}$  and  $T$ . It turns out that in general the causal partition is a coarsening of the observational partition. That is, the causal partition aligns with the observational partition, but the observational partition may subdivide some of the causal classes.

**Theorem 5** (Causal Coarsening). Among all the generative distributions of the form shown in Fig. 2 which induce a given observational partition  $\Pi_o$ , almost all induce a causal partition  $\Pi_c$  that is a coarsening of the  $\Pi_o$ .

Throughout this article, we use “almost all” to mean “all except for a subset of Lebesgue measure zero”. Fig. 3 illustrates the relation between the causal and the observational partition implied by the theorem. We prove the CCT in Supplementary Material A using a technique that extends that of Meek (1995): We show that (1) restricting the space of all the possible  $P(T, H, I)$  to only the distributions compatible with a fixed observational partition puts a linear constraint on the distribution space; (2) requiring that the CCT be false puts a non-trivial polynomial constraint on this subspace, and finally, (3) it follows that the theorem holds for almost all distributions that agree with the given observational partition. The proof strategy indicates a close connection between the CCT and the faithfulness assumption (Spirtes et al., 2000). We note that the measure-zero subset where  $\Pi_c$  does not coarsen  $\Pi_o$  can indeed be non-empty. We provide such counter-examples in Supplementary Material B.

Two points are worth noting here: First, the CCT is interesting inasmuch as the visual causes of a behavior do not contain all the information in the image that predict the behavior. Such information, though not itself a cause of

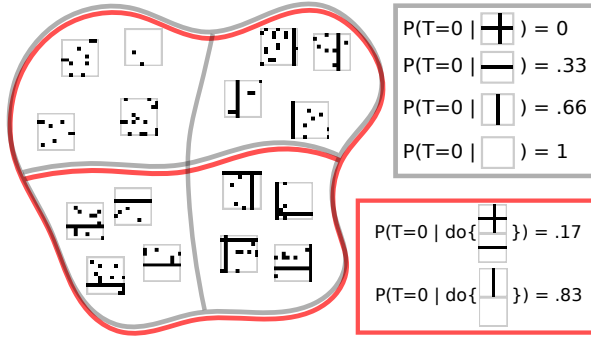


Figure 3: The Causal Coarsening Theorem. The observational probabilities of  $T$  given  $I$  (gray frame) induce an observational partition on the space of all the images (left, observational partition in gray). The causal probabilities (red frame) induce a causal partition, indicated on the left in red. The CCT allows us to expect that the causal partition is a coarsening of the observational partition. The observational and causal probabilities correspond to the generative model shown in Fig. 1.

the behavior, can be informative about the state of other non-visual causes of the target behavior. Second, the CCT allows us to take any classification problem in which the data is divided into observational classes, and assume that the causal labels do not change within each observational class. This will help us develop efficient causal inference algorithms in Section 3.

## 2.4 VISUAL CAUSES IN A CAUSAL MODEL CONSISTING OF MACRO-VARIABLES

We can now simplify our generative model by omitting all the information in  $I$  unrelated to behavior  $T$ . Assume that the observational partition  $\Pi_o^T$  refines the causal partition  $\Pi_c^T$ . Each of the causal classes  $c_1, \dots, c_K$  delineates a region in the image space  $\mathcal{I}$  such that all the images belonging to that region induce the same  $P(T \mid \text{man}(I))$ . Each of those regions—say, the  $k$ -th one—can be further partitioned into sub-regions  $s_1^k, \dots, s_{M_k}^k$  such that all the images in the  $m$ -th sub-region of the  $k$ -th causal region induce the same observational probability  $P(T \mid I)$ . By assumption, the observational partition has a finite number of classes, and we can arbitrarily order the observational classes within each causal class. Once such an ordering is fixed, we can assign an integer  $m \in \{1, 2, \dots, M_k\}$  to each image  $i$  belonging to the  $k$ -th causal class such that  $i$  belongs to the  $m$ -th observational class among the  $M_k$  observational classes contained in  $c_k$ . By construction, this integer explains all the variation of the observational class within a given causal class. This suggests the following definition:

**Definition 6 (Spurious Correlate).** *The spurious correlate  $S$  is a discrete random variable whose value differentiates between the observational classes contained in any causal*

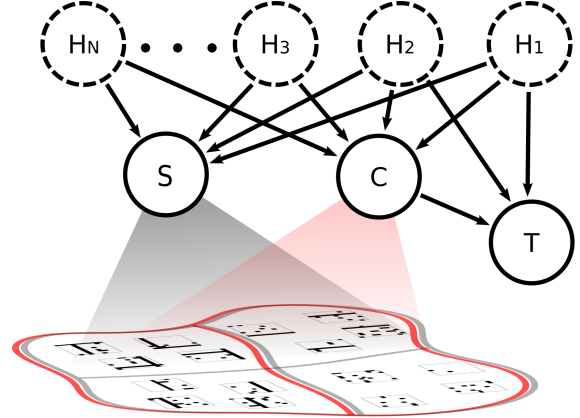


Figure 4: A macro-variable model of visual causation. Using our theory of visual causation we can aggregate the information present in visual micro-variables (image pixels) into the visual cause  $C$  and spurious correlate  $S$ . According to Theorem 7,  $C$  and  $S$  contain all the information about  $T$  available in  $I$ .

class.

The spurious correlate is a well-defined function on  $\mathcal{I}$ , whose value ranges between 1 and  $\max_k M_k$ . Like  $C$ , the spurious correlate  $S$  is a macro-variable constructed from the pixels that make up the image.  $C$  and  $S$  together contain all and only the visual information in  $I$  relevant to  $T$ , but only  $C$  contains the causal information:

**Theorem 7 (Complete Macro-variable Description).** *The following two statements hold for  $C$  and  $S$  as defined above:*

1.  $P(T \mid I) = P(T \mid C, S)$ .
2. Any other variable  $X$  such that  $P(T \mid I) = P(T \mid X)$  has Shannon entropy  $H(X) \geq H(C, S)$ .

We prove the theorem in Supplementary Material C. It guarantees that  $C$  and  $S$  constitute the smallest-entropy macro-variables that encompass all the information about the relationship between  $T$  and  $I$ . Fig. 4 shows the relationship between  $C, S$  and  $T$ , the image space  $\mathcal{I}$  and the observational and causal partitions schematically.  $C$  is now a cause of  $T$ ,  $S$  correlates with  $T$  due to the unobserved common causes  $\mathbf{H}_C$ , and any information irrelevant to  $T$  is pushed into the independent noise variables (commonly not shown in graphical representations of structural equation models).<sup>3</sup>

The macro-variable model lends itself to the standard treatment of causal graphical models described in Pearl

<sup>3</sup>We note that  $C$  may retain predictive information about  $T$  that is not causal, i.e. it is not the case that all spurious correlations can be accounted for in  $S$ . See Supplementary Material D for an example.

(2009). We can define interventions on the causal variables  $\{C, S, T\}$  using the standard *do*-operation. The *do*-operator only sets the value of the intervened variable to the desired value, making it independent of its causes, but it does not (directly) affect the other variables in the system or the relationships between them (see the *modularity assumption* in Pearl (2009)). However, unlike the standard case where causal variables are separated in location (e.g. *smoking* and *lung cancer*), the causal variables in an image may involve the same pixels:  $C$  may be the average brightness of the image, whereas  $S$  may indicate the presence or absence of particular shapes in the image. An intervention on a causal variable using the *do*-operator thus requires that the underlying manipulation of the image respects the state of the other causal variables:

**Definition 8** (Causal Intervention on Macro-variables). *Given the set of macro-variables  $\{C, S\}$  that take on values  $\{c, s\}$  for an image  $i \in \mathcal{I}$ , an intervention  $do(C = c')$  on the macro-variable  $C$  is given by the manipulation of the image  $man(I = i')$  such that  $C(i') = c'$  and  $S(i') = s$ . The intervention  $do(S = s')$  is defined analogously as the change of the underlying image that keeps the value of  $C$  constant.*

In some cases it can be impossible to manipulate  $C$  to a desired value without changing  $S$ . We do not take this to be a problem special to our case. In fact, in the standard macro-variable setting of causal analysis we would expect interventions to be much more restricted by physical constraints than we are with our interventions in the image space.

### 3 CAUSAL FEATURE LEARNING: INFERENCE ALGORITHMS

Given the theoretical specification of the concepts of interest in the previous section, we can now develop algorithms to learn  $C$ , the visual cause of a behavior. In addition, knowledge of  $C$  will allow us to specify a *manipulator function*: a function that, given any image, can return a maximally similar image with the desired causal effect.

**Definition 9** (Manipulator Function). *Let  $C$  be the causal variable of  $T$  and  $d$  a metric on  $\mathcal{I}$ . The manipulator function of  $C$  is a function  $M_C: \mathcal{I} \times \mathcal{C} \rightarrow \mathcal{I}$  such that  $M_C(i, k) = \arg \min_{i \in C^{-1}(k)} d(i, \hat{i})$  for any  $i \in \mathcal{I}, k \in \mathcal{C}$ . In case  $d(i, \cdot)$  has multiple minima, we group them together into one equivalence class and leave the choice of the representative to the manipulator function.*

The manipulator searches for an image closest to  $I$  among all the images with the desired causal effect  $k$ . The meaning of “closest” depends on the metric  $d$  and is discussed further in Section 3.2 below. Note that the manipulator function can find candidates for the image manipulation underlying the desired causal manipulation  $do(C = c)$ , but it does not check whether other variables in the system (in

particular, the spurious correlate) remain in fact unchanged. Using the closest possible image with the desired causal effect is a heuristic approach to fulfilling that requirement.

There are several reasons why we might want such a manipulator function:

- If our goal is to perform causal manipulations on images, the manipulator function offers an automated solution.
- A manipulator that uses a given  $C$  and produces images with the desired causal effect provides strong evidence that  $C$  is indeed the visual cause of the behavior.
- Using the manipulator function we can enrich our dataset with new datapoints, in hope of achieving better generalization on both the causal and predictive learning tasks.

The problem of visual causal feature learning can now be posed as follows: Given an image space  $\mathcal{I}$  and a metric  $d$ , learn  $C$ —the visual cause of  $T$ —and the manipulator  $M_C$ .

#### 3.1 CAUSAL EFFECT PREDICTION

A standard machine learning approach to learning the relation between  $I$  and  $T$  would be to take an *observational dataset*  $\mathcal{D}_{obs} = \{(i_k, P(T | i_k))\}_{k=1, \dots, N}$  and learn a predictor  $f$  whose training performance guarantees a low test error (so that  $f(i^*) \approx P(T | i^*)$  for a test image  $i^*$ ). In causal feature learning, low test error on observational data is insufficient; it is entirely possible that  $\mathcal{D}$  contains spurious information useful in predicting test labels which is nevertheless not causal. That is, the prediction may be highly accurate for observational data, but completely inaccurate for a prediction of the effect of a manipulation of the image (recall the barometer example). However, we can use the CCT to obtain a causal dataset from the observational data, and then train a predictor on that dataset. Algorithm 1 uses this strategy to learn a function  $C$  that, presented with any image  $i \in \mathcal{I}$ , returns  $C(i) \approx P(T | man(I = i))$ . We use a fixed neural network architecture to learn  $C$ , but any differentiable hypothesis class could be substituted instead. Differentiability of  $C$  is necessary in Section 3.2 in order to learn the manipulator function.

In Step 1 the algorithm picks a representative member of each observational class. The CCT tells us that the causal partition coarsens the observational one. That is, in principle (ignoring sampling issues) it is sufficient to estimate  $\hat{C}_m = P(T | man(I = i_{k_m}))$  for just one image in an observational class  $m$  in order to know that  $P(T | man(I = i)) = \hat{C}_m$  for any other  $i$  in the same observational class. The choice of the experimental method of estimating the causal class in Step 2 is left to the user and depends on the behaving agent and the behavior in question. If, for example,  $T$  represents whether the spiking rate of a recorded neuron is above a fixed threshold,

---

**Algorithm 1: Causal Predictor Training**

---

**input** :  $\mathcal{D}_{obs} = \{(i_1, p_1 = p(T | i_1)), \dots, (i_N, p_N = p(T | i_N))\}$  – observational data  
 $\mathcal{P} = \{P_1, \dots, P_M\}$  – the set of observational classes (so that  $\forall k, p_k \in \mathcal{P}, 1 \leq k \leq N$ )  
 $\text{Train}$  – a neural net training algorithm

**output**:  $C: \mathcal{I} \rightarrow [0, 1]$  – the causal variable

- 1 Pick  $\{i_{k_1}, \dots, i_{k_M}\} \subset \{i_1, \dots, i_N\}$  s.t.  $p_{k_m} = P_m$ ;
- 2 Estimate  $\hat{C}_m \leftarrow P(T | \text{man}(I = i_{k_m}))$  for each  $m$ ;
- 3 For all  $k$  let  $\hat{C}(i_k) \leftarrow \hat{C}_m$  if  $p_k = P_m$ ;
- 4  $\mathcal{D}_{csl} \leftarrow \{(i_1, \hat{C}(i_1)), \dots, (i_N, \hat{C}(i_N))\}$ ;
- 5  $C \leftarrow \text{Train}(\mathcal{D}_{csl})$ ;

---

estimating  $P(T | \text{man}(I = i))$  could consist of recording the neuron’s response to  $i$  in a laboratory setting multiple times, and then calculating the probability of spiking from the finite sample. The causal dataset created in Step 4 consists of the observational inputs and their causal classes. The causal dataset is acquired through  $\mathcal{O}(N)$  experiments, where  $N$  is the number of observational classes. The final step of the algorithm trains a neural network that predicts the causal labels on unseen images. The choice of the method of training is again left to the user.

### 3.2 CAUSAL FEATURE MANIPULATION

Once we have learned  $C$  we can use the causal neural network to create synthetic examples of images as similar as possible to the originals, but with a different causal label. The meaning of “as similar as possible” depends on the image metric  $d$  (see Definition 9). The choice of  $d$  is task-specific and crucial to the quality of the manipulations. In our experiments, we use a metric induced by an  $L_2$  norm. Alternatives include other  $L_p$ -induced metrics, distances in implicit feature spaces induced by image kernels (Harchaoui and Bach, 2007; Grauman and Darrell, 2007; Bosch et al., 2007; Vishwanathan, 2010) and distances in learned representation spaces (Bengio et al., 2013).

Algorithm 2 proposes one way to learn the manipulator function using a simple manipulation procedure that approximates the requirements of Definition 9 up to local minima. The algorithm, inspired by the active learning techniques of uncertainty sampling (Lewis and Gale, 1994) and density weighing (Settles and Craven, 2008), starts off by training a causal neural network in Step 2. If only observational data is available, this can be achieved using Algorithm 1. Next, it randomly chooses a set of images to be manipulated, and their target post-manipulation causal labels. The loop that starts in Step 6 then takes each of those images and searches for the image that, among the images with the same desired causal class, is closest to the original image. Note that the causal class boundaries are defined by the current causal neural net  $C$ . Since  $C$  is in general a

---

**Algorithm 2: Manipulator Function Learning**

---

**input** :  $d: \mathcal{I} \times \mathcal{I} \rightarrow \mathbb{R}_+$  – a metric on the image space  
 $\mathcal{D}_{csl} = \{(i_1, c_1), \dots, (i_N, c_N)\}$  – causal data  
 $\mathcal{C} = \{C_1, \dots, C_M\}$  – the set of causal classes (so that  $\forall i, c_i \in \mathcal{C}$ )  
 $\text{Train}$  – a neural net training algorithm  
 $n\text{Iters}$  – number of experiment iterations  
 $Q$  – number of queries per iteration  
 $\alpha$  – manipulation tuning parameter  
 $A: \mathcal{I} \rightarrow \mathcal{C}$  – an oracle for  $P(T | do(I))$

**output**:  $M_C: \mathcal{I} \times \mathcal{C} \rightarrow \mathcal{I}$  – the manipulator function

- 1 **for**  $l \leftarrow 1$  **to**  $n\text{Iters}$  **do**
- 2      $C \leftarrow \text{Train}(\mathcal{D}_{csl})$ ;
- 3     Choose manipulation starting points  $\{i_{l,1}, \dots, i_{l,Q}\}$  at random from  $\mathcal{D}_{csl}$ ;
- 4     Choose manipulation targets  $\{\hat{c}_{l,1}, \dots, \hat{c}_{l,Q}\}$  such that  $\hat{c}_{l,k} \neq c_{l,k}$ ;
- 5     **for**  $k \leftarrow 1$  **to**  $Q$  **do**
- 6          $\hat{i}_{l,k} \leftarrow \operatorname{argmin}_{j \in \mathcal{I}} (1 - \alpha)|C(j) - \hat{c}_{l,k}| + \alpha d(j, i_{l,k})$ ;
- 7     **end**
- 8      $\mathcal{D}_{csl} \leftarrow \mathcal{D}_{csl} \cup \{(i_{l,1}, A(\hat{i}_{l,1})), \dots, (i_{l,Q}, A(\hat{i}_{l,Q}))\}$ ;
- 9 **end**

---

highly nonlinear function and it can be hard to find its inverse sets, we use an approximate solution. The algorithm thus finds the minimum of a weighted sum of  $|C(j) - \hat{c}_{l,k}|$  (the difference of the output image  $j$ ’s label and the desired label  $\hat{c}_{l,k}$ ) and  $d(i_{l,k}, j)$  (the distance of the output image  $j$  from the original image  $i_{l,k}$ ).

At each iteration, the algorithm performs  $Q$  manipulations and the same number of causal queries to the agent, which result in new datapoints  $(\hat{i}_{l,1}, A(\hat{i}_{l,1})), \dots, (\hat{i}_{l,Q}, A(\hat{i}_{l,Q}))$ . It is natural to claim that the manipulator performs well if  $A(\hat{i}_{l,k}) \approx \hat{c}_{l,k}$  for many  $k$ , which means the target causal labels agree with the true causal labels. We thus define the *manipulation error* of the  $l$ th iteration  $M\text{Err}_l$  as

$$M\text{Err}_l = \frac{1}{Q} \sum_{k=1}^Q |A(\hat{i}_{l,k}) - \hat{c}_{l,k}|. \quad (2)$$

While it is important that our manipulations are accurate, we also want them to be minimal. Another measure of interest is thus the *average manipulation distance*

$$M\text{Dist}_l = \frac{1}{Q} \sum_{k=1}^Q d(I_{l,k}, \hat{i}_{l,k}). \quad (3)$$

A natural variant of Algorithm 2 is to set  $n\text{Iters}$  to a large

integer and break the loop when one or both of these performance criteria reaches a desired value.

## 4 EXPERIMENTS

In order to illustrate the concepts presented in this article we perform two causal feature learning experiments. The first experiment, called GRATING, uses observational and causal data generated by the model from Section 1.2. The GRATING experiment confirms that our system can learn the ground truth cause and ignore the spurious correlates of a behavior. The second experiment, MNIST, uses images of hand-written digits (LeCun et al., 1998) to exemplify the use of the manipulator function on slightly more realistic data: in this example, we transform an image into a maximally similar image with another class label.

We chose problems that are simple from the computer vision point of view. Our goal is to develop the theory of visual causal feature learning and show that it has feasible algorithmic solutions; we are at this point not engineering advanced computer vision systems.

### 4.1 THE GRATING EXPERIMENT

In this experiment we generate data using the model of Fig. 1, with two minor differences:  $H_1$  and  $H_2$  only induce one v-bar or h-bar in the image and we restrict our observational dataset to images with only about 3% of the pixels filled with random noise (see Fig. 5). Both restrictions increase the clarity of presentation. We use Algorithms 1 and 2 (with minor modifications imposed by the binary nature of the images) to learn the visual cause of behavior  $T$ .

Figure 5 (top) shows the progress of the training process. The first step (not shown in the figure) uses the CCT to learn the causal labels on the observational data. We then train a simple neural network (a fully connected network with one hidden layer of 100 units) on this data. The same network is used on Iteration 1 to create new manipulated exemplars. We then follow Algorithm 2 to train the manipulator iteratively. Fig. 5 (bottom) illustrates the difference between the manipulator on Iteration 1 (which fails almost 40% of the time) and Iteration 20, where the error is about 6%. Each column shows example manipulations of a particular kind. Columns with green labels indicate successful manipulations of which there are two kinds: switching the causal variable on ( $0 \Rightarrow 1$ , “adding the h-bar”), or switching it off ( $1 \Rightarrow 0$ , “removing the h-bar”). Red-labeled columns show cases in which the manipulator failed to influence the cause: That is, each red column shows an original image and its manipulated version which the manipulator believes should cause a change in  $T$ , but which does not induce such change. The red/green horizontal bars show the percentage of success/error for each manipulation direction. Fig. 5 (bottom, a) shows that after training on the

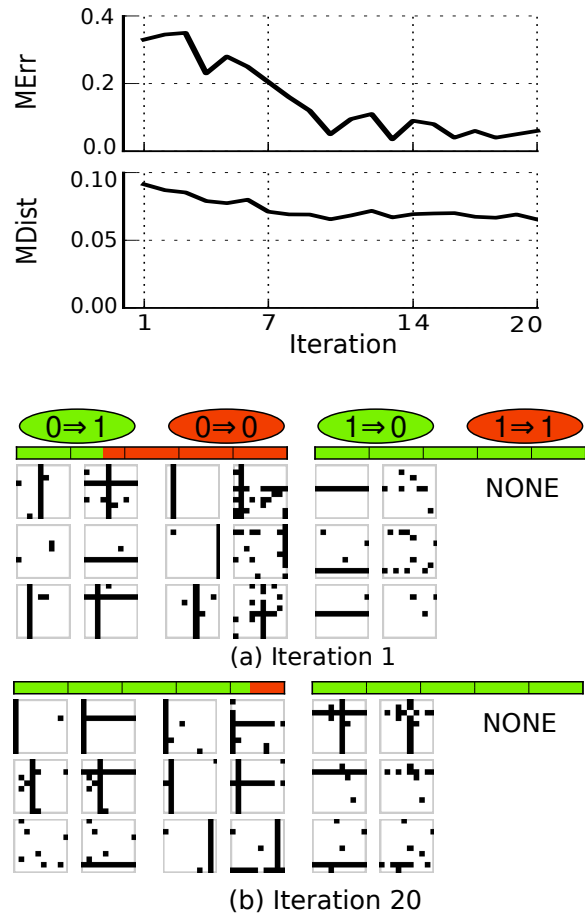


Figure 5: Manipulator learning for GRATING. **Top.** The plots show the progress of our manipulator function learning algorithm over twenty iterations of experiments for the GRATING problem. The manipulation error decreases quickly with progressing iterations, whereas the manipulation distance stays close to constant. **Bottom.** Original and manipulated GRATING images. See text for the details.

causally-coarsened observational dataset, the manipulator fails about 40% of the time. In Fig. 5 (b), after twenty manipulator learning iterations, only six manipulations out of a hundred are unsuccessful. Furthermore, the causally irrelevant image pixels are also much better preserved than at iteration 1. The fully-trained manipulator correctly learned to manipulate the presence of the h-bar to cause changes in  $T$ , and ignores the v-bar that is strongly correlated with the behavior but does not cause it.

### 4.2 THE MNIST ON MTURK EXPERIMENT

In this experiment we start with the MNIST dataset of handwritten digits. In our terminology, this – as well as any standard vision dataset – is already causal data: the labels are assigned in an experimental setting, not “in nature”.

Consider the following binary human behavior:  $T = 1$  if a human observer answers affirmatively to the question

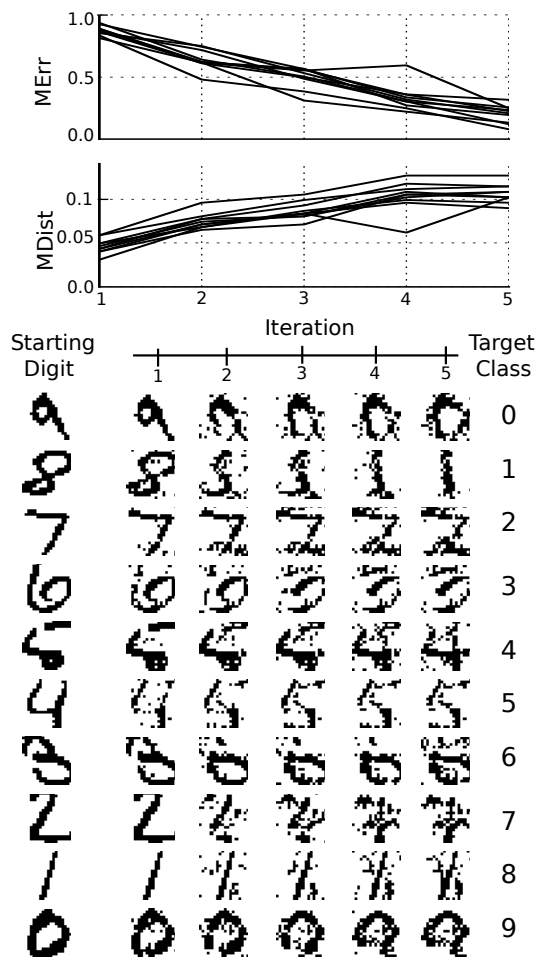


Figure 6: Manipulator Learning for MNIST ON MTURK. **Top.** In contrast to the GRATING experiment, here the manipulation distance grows as the manipulation error decreases. This is because a successful manipulator needs to change significant parts of each image (such as continuous strokes). **Bottom.** Visualization of manipulator training on randomly selected (not cherry-picked) MNIST digits. See text for the details.

“Does this image contain the digit ‘7’?”, while  $T = 0$  if the observer judges that the image does not contain the digit ‘7’. For simplicity we will assume that for any image either  $P(T = 1 | man(I)) = 0$  or  $P(T = 1 | man(I)) = 1$ . Our task is to learn the manipulator function that will take any image and modify it minimally such that it will become a ‘7’ if it was not before, or will stop resembling a ‘7’ if it did originally.

We conduct the manipulator training separately for all the ten MNIST digits using human annotators on Amazon Mechanical Turk. The exact training procedure is described in Supplementary Material E. Fig. 6 (top) shows training progress. As in Fig. 5, the manipulation error decreases with training. Fig. 6 (bottom) visualizes the manipulator training progress. In the first row we see a randomly chosen MNIST “9” being manipulated to resemble a “0”, pushed

through successive “0-vs-all” manipulators trained at iterations 0, 1, ..., 5 (iteration 1 shows what the neural net takes to be the closest manipulation to change the “9” to a “0” purely on the basis of the non-manipulated data). Further rows perform similar experiments for the other digits. The plots show how successive manipulators progressively remove the original digits’ features and add target class features to the image.

## 5 DISCUSSION

We provide a link between causal reasoning and neural network models that have recently enjoyed tremendous success in the fields of machine learning and computer vision (LeCun et al., 1998; Russakovsky et al., 2014). Despite very encouraging results in image classification (Krizhevsky et al., 2012), object detection (Dollar et al., 2012) and fine-grained classification (Branson et al., 2014; Zhang et al., 2014), some researchers have found that visual neural networks can be easily fooled using adversarial examples (Szegedy et al., 2014; Goodfellow et al., 2014). The learning procedure for our manipulator function could be viewed as an attempt to train a classifier that is robust against such examples. The procedure uses causal reasoning to improve on the boundaries of a standard, correlational classifier (Fig. 5 and 6 show the improvement). However, the ultimate purpose of a causal manipulator network is to extract truly causal features from data and automatically perform causal manipulations based on those features.

A second contribution concerns the field of causal discovery. Modern causal discovery algorithms presuppose that the set of causal variables is well-defined and meaningful. What exactly this presupposition entails is unclear, but there are clear counter-examples:  $x$  and  $2x$  cannot be two distinct causal variables. There are also well understood problems when causal variables are aggregates of other variables (Chu et al., 2003; Spirtes and Scheines, 2004). We provide an account of how causal macro-variables can supervene on micro-variables.

This article is an attempt to clarify how one may construct a set of well-defined causal macro-variables that function as basic relata in a causal graphical model. This step strikes us as essential if causal methodology is to be successful in areas where we do not have clearly delineated candidate causes or where causes supervene on micro-variables, such as in climate science and neuroscience, economics and—in our specific case—vision.

### Acknowledgements

KC’s work was funded by the Qualcomm Innovation Fellowship 2014. KC’s and PP’s work was supported by the ONR MURI grant N00014-10-1-0933. FE would like to thank Cosma Shalizi for pointers to many relevant results this paper builds on.

## References

- Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.
- A. Bosch, A. Zisserman, and X. Munoz. Representing shape with a spatial pyramid kernel. In *6th ACM International Conference on Image and Video Retrieval*, pages 401–408, 2007.
- S. Branson, G. Van Horn, and C. Wah. The Ignorant Led by the Blind: A Hybrid Human–Machine Vision System for Fine-Grained Categorization. *International Journal of Computer Vision*, 108(1-2):3–29, 2014.
- T. Chu, C. Glymour, R. Scheines, and P. Spirtes. A statistical problem for inference to regulatory structure from associations of gene expression measurements with microarrays. *Bioinformatics*, 19(9):1147–1152, 2003.
- P. Dollar, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: An evaluation of the state of the art. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4):743–761, 2012.
- A. S. Fire and S. C. Zhu. Using causal induction in humans to learn and infer causality from video. *The Annual Meeting of the Cognitive Science Society (CogSci)*, 2013a.
- A. S. Fire and S. C. Zhu. Learning Perceptual Causality from Video. *AAAI Workshop: Learning Rich Representations from Low-Level Sensors*, 2013b.
- I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and Harnessing Adversarial Examples. *arXiv preprint arXiv:1412.6572*, 2014.
- K. Grammer and R. Thornhill. Human (*Homo sapiens*) facial attractiveness and sexual selection: The role of symmetry and averageness. *Journal of Comparative Psychology*, 108(3):233–242, 1994.
- K. Grauman and T. Darrell. The pyramid match kernel: Efficient learning with sets of features. *Journal of Machine Learning Research*, 8:725–260, 2007.
- I. Guyon, A. Elisseeff, and C. Aliferis. Causal feature selection. In *Computational Methods of Feature Selection Data Mining and Knowledge Discovery Series*, pages 63–85. Chapman and Hall/CRC, 2007.
- Z. Harchaoui and F. Bach. Image classification with segmentation graph kernels. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.
- E. P. Hoel, L. Albantakis, and G. Tononi. Quantifying causal emergence shows that macro can beat micro. *Proceedings of the National Academy of Sciences*, 110(49):19790–19795, 2013.
- A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. 2012.
- Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- D. D. Lewis and W. A. Gale. A sequential algorithm for training text classifiers. In *ACM SIGIR Seventeenth Conference on Research and Development in Information Retrieval*, pages 3–12, 1994.
- C. Meek. Strong completeness and faithfulness in Bayesian networks. In *Eleventh Conference on Uncertainty in Artificial Intelligence*, pages 411–418, 1995.
- J. Pearl. *Causality: Models, Reasoning and Inference*. Cambridge University Press, 2000.
- J. P. Pellet and A. Elisseeff. Using Markov blankets for causal structure learning. *Journal of Machine Learning Research*, 9:1295–1342, 2008.
- O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet large scale visual recognition challenge. *arXiv preprint arXiv:1409.0575*, 2014.
- B. Settles and M. Craven. An analysis of active learning strategies for sequence labeling tasks. In *Conference on Empirical Methods in Natural Language Processing*, pages 1070–1079, 2008.
- C. R. Shalizi. *Causal architecture, complexity and self-organization in the time series and cellular automata*. PhD thesis, University of Wisconsin at Madison, 2001.
- C. R. Shalizi and J. P. Crutchfield. Computational mechanics: Pattern and prediction, structure and simplicity. *Journal of Statistical Physics*, 104(3-4):817–879, 2001.
- P. Spirtes and R. Scheines. Causal inference of ambiguous manipulations. *Philosophy of Science*, 71(5):833–845, 2004.
- P. Spirtes, C. N. Glymour, and R. Scheines. *Causation, prediction, and search*. Massachusetts Institute of Technology, 2nd ed. edition, 2000.
- C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014.
- S. V. N. Vishwanathan. Graph kernels. *Journal of Machine Learning Research*, 11:1201–1242, 2010.
- N. Zhang, J. Donahue, R. Girshick, and T. Darrell. Part-based R-CNNs for fine-grained category detection. In *ECCV 2014*, pages 834–849, 2014.



---

# Large-Margin Determinantal Point Processes

---

**Wei-Lun Chao\***  
U. of Southern California  
Los Angeles, CA 90089  
weilunc@usc.edu

**Boqing Gong\***  
U. of Southern California  
Los Angeles, CA 90089  
boqinggo@usc.edu

**Kristen Grauman**  
U. of Texas at Austin  
Austin, TX 78701  
grauman@cs.utexas.edu

**Fei Sha**  
U. of Southern California  
Los Angeles, CA 90089  
feisha@usc.edu

## Abstract

Determinantal point processes (DPPs) offer a powerful approach to modeling diversity in many applications where the goal is to select a diverse subset from a ground set of items. We study the problem of learning the parameters (i.e., the kernel matrix) of a DPP from labeled training data. In this paper, we develop a novel parameter estimation technique particularly tailored for DPPs based on the principle of large margin separation. In contrast to the state-of-the-art method of maximum likelihood estimation of the DPP parameters, our large-margin loss function explicitly models errors in selecting the target subsets, and it can be customized to trade off different types of errors (precision vs. recall). Extensive empirical studies validate our contributions, including applications on challenging document and video summarization, where flexibility in balancing different errors while training the summarization models is indispensable.

## 1 INTRODUCTION

Imagine we are to design a search engine to retrieve web images that match user queries. In response to the search term JAGUAR, what should we retrieve—the images of the animal jaguar or the images of the automobile jaguar?

This frequently cited example illustrates the need to incorporate the notion of *diversity*. In many tasks, we want to select a subset of items from a “ground set”. While the ground set might contain many similar items, our goal is *not* to discover all of the same ones, but rather to find a subset of diverse items that ensure coverage (the exact definition of coverage is task-specific). In the example of retrieving images for JAGUAR, we achieve diversity by including both types of images.

---

\*Equal contribution

Recently, the determinantal point process (DPP) has emerged as a promising technique for modeling diversity [Kulesza and Taskar, 2012]. A DPP defines a probability distribution over the power set of a ground set. Intuitively, subsets of higher diversity are assigned larger probabilities, and thus are more likely to be selected than those with lower diversity. Since its original application to quantum physics, DPP has found many applications in modeling random trees and graphs [Burton and Pemantle, 1993], document summarization [Kulesza and Taskar, 2011b], search and ranking in information retrieval [Kulesza and Taskar, 2011a], and clustering [Kang, 2013]. Various extensions have also been studied, including k-DPP [Kulesza and Taskar, 2011a], structured DPP [Kulesza and Taskar, 2011c], Markov DPP [Affandi et al., 2012], and DPP on continuous spaces [Affandi et al., 2013].

The probability distribution of a DPP depends crucially on its kernel—a square and symmetric, positive semidefinite matrix whose elements specify how similar every pair of items in the ground set are. This kernel matrix is often unknown and needs to be estimated from training data.

This is a very challenging problem for several reasons. First, the number of the parameters, i.e., the number of elements in the kernel matrix, is quadratic in the number of items in the ground set. For many tasks (for instance, image search), the ground set can be very large. Thus it is impractical to directly specify every element of the matrix, and a suitable reparameterization of the matrix is necessary. Secondly, the number of training samples is often limited in many practical applications. One such example is the task of document summarization, where our aim is to select a succinct subset of sentences from a long document. There, acquiring accurate annotations from human experts is costly and difficult. Thirdly, for many tasks, we need to evaluate the performance of the learned DPP not only by its accuracy in predicting whether an item should be selected, but also by other measures like precision and recall. For instance, failing to select key sentences for summarizing documents might be regarded as being more catastrophic than injecting sentences with repetitive information into the

summary.

Existing methods of parameter estimation for DPPs are inadequate to address these challenges. For example, maximum likelihood estimation (MLE) typically requires a large number of training samples in order to estimate the underlying model correctly. This also limits the number of the parameters it can estimate reliably, restricting its use to DPPs whose kernels can be parameterized with few degrees of freedom. It also does not offer fine control over precision and recall.

We propose a two-pronged approach for learning a DPP from labeled data. First, we improve modeling flexibility by reparameterizing the DPP’s kernel matrix with multiple base kernels. This representation could easily incorporate domain knowledge and requires learning fewer parameters (instead of the whole kernel matrix). Then, we optimize the parameters such that the probability of the correct subset is larger than other erroneous subsets by a large margin. This margin is task-specific and can be customized to reflect the desired performance measure—for example, to monitor precision and recall. As such, our approach defines objective functions that closely track selection errors and work well with few training samples. While the principle of large margin separation has been widely used in classification [Vapnik, 1998] and structured prediction [Taskar et al., 2005], formulating DPP learning with the large margin principle is novel. Our empirical studies show that the proposed method attains superior performance on two challenging tasks of practical interest: document and video summarization.

The rest of the paper is organized as follows. We provide background on the DPP in section 2, followed by our approach in section 3. We discuss related work in section 4 and report our empirical studies in section 5. We conclude in section 6.

## 2 BACKGROUND: DETERMINANTAL POINT PROCESSES

We first review background on the determinantal point process (DPP) [Macchi, 1975] and the standard maximum likelihood estimation technique for learning DPP parameters from data. More details can be found in the excellent tutorial [Kulesza and Taskar, 2012].

Given a ground set of  $M$  items,  $\mathcal{Y} = \{1, 2, \dots, M\}$ , a DPP defines a probabilistic measure over the power set, i.e., all possible subsets (including the empty set) of  $\mathcal{Y}$ . Concretely, let  $\mathbf{L}$  denote a symmetric and positive semidefinite matrix in  $\mathbb{R}^{M \times M}$ . The probability of selecting a subset  $\mathbf{y} \subseteq \mathcal{Y}$  is given by

$$P(\mathbf{y}; \mathbf{L}) = \det(\mathbf{L} + \mathbf{I})^{-1} \det(\mathbf{L}_{\mathbf{y}}), \quad (1)$$

where  $\mathbf{L}_{\mathbf{y}}$  denotes the submatrix of  $\mathbf{L}$ , with rows and

columns selected by the indices in  $\mathbf{y}$ .  $\mathbf{I}$  is the identity matrix with the proper size. We define  $\det(\mathbf{L}_{\emptyset}) = 1$ . The above way of defining a DPP is called an L-ensemble. An equivalent way of defining a DPP is to use a kernel matrix to define the marginal probability of selecting a random subset:

$$P_{\mathbf{y}} = \sum_{\mathbf{y}' \subseteq \mathcal{Y}} P(\mathbf{y}'; \mathbf{L}) \mathbb{I}[\mathbf{y} \subseteq \mathbf{y}'] = \det(\mathbf{K}_{\mathbf{y}}), \quad (2)$$

where we sum over all subsets  $\mathbf{y}'$  that contain  $\mathbf{y}$  ( $\mathbb{I}[\cdot]$  is an indicator function). The matrix  $\mathbf{K}$  is another positive semidefinite matrix, computable from the  $\mathbf{L}$  matrix

$$\mathbf{K} = \mathbf{L}(\mathbf{L} + \mathbf{I})^{-1}, \quad (3)$$

and  $\mathbf{K}_{\mathbf{y}}$  is the submatrix of  $\mathbf{K}$  indexed by  $\mathbf{y}$ . Despite the exponential number of summands in eq. (2), the marginalization is analytically tractable and computable in polynomial time.

### 2.1 MODELING DIVERSITY

One particularly useful property of the DPP is its ability to model *pairwise repulsion*. Consider the marginal probability of having two items  $i$  and  $j$  simultaneously in a subset:

$$\begin{aligned} P_{\{i,j\}} &= \det \begin{vmatrix} K_{ii} & K_{ij} \\ K_{ji} & K_{jj} \end{vmatrix} = K_{ii}K_{jj} - K_{ij}^2 \\ &\leq K_{ii}K_{jj} = P_{\{i\}}P_{\{j\}} \leq \min(P_{\{i\}}, P_{\{j\}}). \end{aligned} \quad (4)$$

Thus, unless  $K_{ij} = 0$ , the probability of observing  $i$  and  $j$  jointly is always less than observing either  $i$  or  $j$  separately. Namely, having  $i$  in a subset repulsively excludes  $j$  and vice versa. Another extreme case is when  $i$  and  $j$  are the same; then  $K_{ii} = K_{jj} = K_{ij}$ , which leads to  $P_{\{i,j\}} = 0$ . Namely, we should never allow them together in any subset.

Consequently, a subset with a large (marginal) probability cannot have too many items that are similar to each other (i.e., with high values of  $K_{ij}$ ). In other words, the probability provides a gauge of the diversity of the subset. The most diverse subset, which balances all the pairwise repulsions, is the subset that attains the highest probability

$$\mathbf{y}^{\text{MAP}} = \arg \max_{\mathbf{y}} P(\mathbf{y}; \mathbf{L}). \quad (5)$$

Note that this MAP inference is computed with respect to the L-ensemble (instead of  $\mathbf{K}$ ) as we are interested in the mode, not the marginal probability of having the subset. Unfortunately, the MAP inference is NP-hard [Ko et al., 1995]. Various approximation algorithms have been investigated [Gillenwater et al., 2012, Kulesza and Taskar, 2012].

## 2.2 MAXIMUM LIKELIHOOD ESTIMATION

Suppose we are given a training set  $\{(\mathcal{Y}_n, \mathbf{y}_n)\}$ , where each ground set  $\mathcal{Y}_n$  is annotated with its most diverse subset  $\mathbf{y}_n$ . How can we discover the underlying parameters  $\mathbf{L}$  or  $\mathbf{K}$ ? Note that different ground sets need not have overlap. Thus, directly specifying kernel values for every pair of items is unlikely to be scalable. Instead, we will need to assume that either  $\mathbf{L}$  or  $\mathbf{K}$  for each ground set is represented by a shared set of parameters  $\theta$ .

For items  $i$  and  $j$  in  $\mathcal{Y}_n$ , suppose their kernel values  $K_{n_{ij}}$  can be computed as a function of  $\mathbf{x}_{n_i}$ ,  $\mathbf{x}_{n_j}$  and  $\theta$ , where  $\mathbf{x}_{n_i}$  and  $\mathbf{x}_{n_j}$  are features characterizing those items. Our learning objective is to optimize  $\theta$  such that  $\mathbf{y}_n$  is the most diverse subset in  $\mathcal{Y}_n$ , or attains the highest probability. This gives rise to the following maximum likelihood estimate (MLE) [Kulesza and Taskar, 2011b],

$$\theta^{\text{MLE}} = \arg \max_{\theta} \sum_n \log P(\mathbf{y}_n; \mathbf{L}_n(\mathcal{Y}_n; \theta)), \quad (6)$$

where  $\mathbf{L}_n(\mathcal{Y}_n; \theta)$  converts features in  $\mathcal{Y}_n$  to the  $\mathbf{L}$  matrix for the ground set  $\mathcal{Y}_n$ . MLE has been a standard approach for estimating DPP parameters. However, as we will discuss in section 3.2, it has important limitations.

Next, we introduce our method for learning the parameters. We first present our multiple kernel based representation of the  $\mathbf{L}$  matrix and then the large-margin based estimation.

## 3 OUR APPROACH

Our approach consists of two components that are developed in parallel, yet work in concert: (1) the use of multiple kernel functions to represent the DPP; (2) applying the principle of large margin separation to optimize the parameters. The former reduces the number of parameters to learn and thus is especially advantageous when the number of training samples is limited. The latter strengthens the advantage by optimizing objective functions that closely track subset selection errors.

### 3.1 MULTIPLE KERNEL REPRESENTATION

Learning the  $\mathbf{L}$  or  $\mathbf{K}$  matrix for a DPP is an instance of learning kernel functions, as those matrices are positive semidefinite matrices, interpretable as kernel functions being evaluated on the items in the ground set. Thus, our goal is essentially to learn the right kernel function to measure similarity.

However, for many applications, similarity is just one of the criteria for selecting items. For instance, in the previous example of image retrieval, the retrieved images not only need to be diverse (thus different) but also need to have strong relevance to the query term. Similarly, in document summarization, the selected sentences not only need to be

succinct and not redundant, but also need to represent the contents of the document [Lin and Bilmes, 2010].

Kulesza and Taskar [2011b] propose to balance these two potentially conflicting forces with a decomposable  $\mathbf{L}$  matrix:

$$\begin{aligned} L_{ij} &= q_i q_j S_{ij} = q_i q_j \phi_i^T \phi_j, \\ q_i &= q(\mathbf{x}_i) = \exp(\theta^T \mathbf{x}_i), \quad \forall i, j \in \mathcal{Y}, \end{aligned} \quad (7)$$

where  $q_i$  is referred to as the *quality* factor, modeling how representative or relevant the selected items are. It depends on item  $i$ 's feature vector  $\mathbf{x}_i$ , which encodes  $i$ 's contextual information and its *representativeness* of other items. For example, in document summarization, possible features are the sentence lengths, positions of the sentences in the text, or others.  $S_{ij}$ , on the other hand, measures how *similar* two sentences are, computed from a different set of features,  $\phi_i$  and  $\phi_j$ , such as bag-of-words descriptors that represent each item's individual characteristics.

However, prior work [Kulesza and Taskar, 2011b] does not investigate whether this specific definition of similarity could be made optimal and adapted to the data, thus limiting the modeling power of the DPP largely to infer the quality  $q_i$ . Our empirical studies show that this limitation can be severe, especially when the modeling choice is erroneous (cf. section 5.2).

In this paper, we retain the aspect of quality modeling but improve the modeling of similarity  $S_{ij}$  in two ways. First, we use nonlinear kernel functions such as the Gaussian RBF kernel to determine similarity. Secondly, and more importantly, we combine several base kernels:

$$S_{ij} = \sum_k \alpha_k \exp\{-\|\phi_i - \phi_j\|_2^2 / \sigma_k^2\} + \beta \phi_i^T \phi_j, \quad (8)$$

where  $k$  indexes the base kernels and  $\sigma_k$  is a scaling factor. The combination coefficients are constrained such that  $\sum_k \alpha_k + \beta = 1$ . They are optimized on the annotated data, either via maximum likelihood estimation or via our novel parameter estimation technique, to be described next.

### 3.2 LARGE-MARGIN ESTIMATION OF DPP

Maximum likelihood estimation does not closely track discriminative errors [Ng and Jordan, 2002, Vapnik, 1998, Jebara, 2004]. While improving the likelihood of the ground-truth subset  $\mathbf{y}_n$ , MLE could also improve the likelihoods of other competing subsets. Consequentially, a model learned with MLE could have modes that are very different subsets yet are very close to each other in their probability values. Having highly confusable modes is especially problematic for DPP's NP-hard MAP inference—the difference between such modes can fall within the approximation errors of approximate inference algorithms such that the true MAP cannot be easily extracted.

### 3.2.1 Multiplicative Large Margin Constraints

To address these deficiencies, our large-margin based approach aims to maintain or increase the margin between the correct subset and alternative, incorrect ones. Specifically, we formulate the following large margin constraints

$$\begin{aligned} \log P(\mathbf{y}_n; \mathbf{L}_n) &\geq \max_{\mathbf{y} \subseteq \mathcal{Y}_n} \log \ell(\mathbf{y}_n, \mathbf{y}) P(\mathbf{y}; \mathbf{L}_n) \\ &= \max_{\mathbf{y} \subseteq \mathcal{Y}_n} \log \ell(\mathbf{y}_n, \mathbf{y}) + \log P(\mathbf{y}; \mathbf{L}_n), \end{aligned} \quad (9)$$

where  $\ell(\mathbf{y}_n, \mathbf{y})$  is a loss function measuring the discrepancy between the correct subset and an alternative  $\mathbf{y}$ . We assume  $\ell(\mathbf{y}_n, \mathbf{y}_n) = 0$ .

Intuitively, the more different  $\mathbf{y}$  is from  $\mathbf{y}_n$ , the larger the gap we want to maintain between the two probabilities. This way, the incorrect one has less chance to be identified as the most diverse one. Note that while similar intuitions have been explored in multi-way classification and structured prediction, the margin here is *multiplicative* instead of additive—this is by design, as it leads to a tractable optimization over the exponential number of constraints, as we will explain later.

### 3.2.2 Design of the Loss Function

A natural choice for the loss function is the Hamming distance between  $\mathbf{y}_n$  and  $\mathbf{y}$ , counting the number of disagreements between two subsets:

$$\ell_H(\mathbf{y}_n, \mathbf{y}) = \sum_{i \in \mathbf{y}} \mathbb{I}[i \notin \mathbf{y}_n] + \sum_{i \notin \mathbf{y}} \mathbb{I}[i \in \mathbf{y}_n]. \quad (10)$$

In this loss function, failing to select the right item costs the same as adding an unnecessary item. In many tasks, however, this symmetry does not hold. For example, in summarizing a document, omitting a key sentence has more severe consequences than adding a (trivial) sentence.

To balance these two types of errors, we introduce the generalized Hamming loss function,

$$\ell_\omega(\mathbf{y}_n, \mathbf{y}) = \sum_{i \in \mathbf{y}} \mathbb{I}[i \notin \mathbf{y}_n] + \omega \sum_{i \notin \mathbf{y}} \mathbb{I}[i \in \mathbf{y}_n]. \quad (11)$$

When  $\omega$  is greater than 1, the learning biases towards higher *recall* to select as many items in  $\mathbf{y}_n$  as possible. When  $\omega$  is significantly less than 1, the learning biases towards high *precision* to avoid incorrect items as much as possible. Our empirical studies demonstrate such flexibility and its advantages in two real-world summarization tasks.

### 3.2.3 Numerical Optimization

To overcome the challenge of dealing with an exponential number of constraints in eq. (9), we reformulate it as a tractable optimization problem. We first upper-bound

the hard-max operation with Jensen’s inequality (i.e., softmax):

$$\begin{aligned} \log P(\mathbf{y}_n; \mathbf{L}_n) &\geq \log \sum_{\mathbf{y} \subseteq \mathcal{Y}} e^{\log \ell_\omega(\mathbf{y}_n, \mathbf{y}) P(\mathbf{y}; \mathbf{L}_n)} \\ &= \text{softmax}_{\mathbf{y} \subseteq \mathcal{Y}_n} \log \ell_\omega(\mathbf{y}_n, \mathbf{y}) + \log P(\mathbf{y}; \mathbf{L}_n). \end{aligned} \quad (12)$$

With the loss function  $\ell_\omega(\mathbf{y}_n, \mathbf{y})$ , the right-hand-side is computable in polynomial time,

$$\begin{aligned} &\text{softmax}_{\mathbf{y} \subseteq \mathcal{Y}_n} \log \ell_\omega(\mathbf{y}_n, \mathbf{y}) + \log P(\mathbf{y}; \mathbf{L}_n) \\ &= \log \left( \sum_{i \notin \mathbf{y}_n} K_{n_{ii}} + \omega \sum_{i \in \mathbf{y}_n} (1 - K_{n_{ii}}) \right), \end{aligned} \quad (13)$$

where  $K_{n_{ii}}$  is the  $i$ -th element on the diagonal of  $\mathbf{K}_n$ , the marginal kernel matrix corresponding to  $\mathbf{L}_n$ . The detailed derivation of this result is in the supplementary material. Note that  $\mathbf{K}_n$  can be computed efficiently from  $\mathbf{L}_n$  through the identity eq. (3).

The softmax can be seen as a summary of all undesirable subsets (the correct subset  $\mathbf{y}_n$  does not contribute to the weighted sum as  $\ell_\omega(\mathbf{y}_n, \mathbf{y}_n) = 0$ ). Our optimization balances this term with the likelihood of the target with the hinge loss function  $[z]_+ = \max(0, z)$ ,

$$\begin{aligned} \min \sum_n &\left[ -\log P(\mathbf{y}_n; \mathbf{L}_n) \right. \\ &\left. + \lambda \log \left( \sum_{i \notin \mathbf{y}_n} K_{n_{ii}} + \omega \sum_{i \in \mathbf{y}_n} (1 - K_{n_{ii}}) \right) \right]_+, \end{aligned} \quad (14)$$

where  $\lambda \geq 0$  is a tradeoff coefficient, to be tuned on validation datasets. Note that this objective function subsumes maximum likelihood estimation where  $\lambda = 0$ . We optimize the objective function with subgradient descent. Details are in the supplementary material.

## 4 RELATED WORK

The DPP arises from random matrix theory and quantum physics [Macchi, 1975, Kulesza and Taskar, 2012]. In machine learning, researchers have proposed different variations to improve its modeling capacity. Kulesza and Taskar [2011a] introduced k-DPP to restrict the sets to have a constant size  $k$ . Affandi et al. [2012] proposed a Markov DPP which offers diversity at adjacent time stamps. A structured DPP was presented in [Kulesza and Taskar, 2011c] to model trees and graphs. The MAP inference of DPP is generally NP-hard [Ko et al., 1995]. Gillenwater et al. [2012] developed an 1/4-approximation algorithm. In practice, greedy inference gives rise to decent results [Kulesza and Taskar, 2011b] though it lacks theoretical guarantees.

Another popular alternative is to resort to fast sampling algorithms [Kang, 2013, Kulesza and Taskar, 2012].

In spite of much research activity surrounding DPPs, there is very little work exploring how to effectively learn the model parameters. MLE is the most popular estimator. Compared to MLE, our approach is more robust to the number of training data or mis-specified models, and offers greater flexibility by incorporating customizable error functions. A recent Bayesian approach works with the posterior over the parameters [Affandi et al., 2014]. In contrast to that work, we develop a large-margin training approach for DPPs and directly minimize the set selection errors. The large margin principle has been widely used in classification [Vapnik, 1998] and structured prediction [Taskar et al., 2005, Tsochantaridis et al., 2004, Taskar et al., 2004, Sha and Saul, 2006], but its application to DPP is original. In order to make it tractable for DPPs, we use multiplicative rather than additive margin constraints.

## 5 EXPERIMENTS

We validate our large-margin approach to learn DPP parameters (DPP<sup>LME</sup>) with extensive empirical studies on both synthetic data and two real-world summarization tasks with documents and videos. While DPP also has applications beyond summarization, this is a particularly good testbed to illustrate diverse subset selection: a compact summary ought to include high quality items that, taken together, offer good coverage of the source content.

### 5.1 SETUP

#### 5.1.1 Evaluation Metrics

We evaluate the quality of the selected subset  $\mathbf{y}^{\text{MAP}}$  against the ground-truth  $\mathbf{y}^*$  using the F-score, which is the harmonic mean of precision and recall:

$$\text{F-score} = \frac{2 \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}},$$

$$\text{Precision} = \frac{|\mathbf{y}^{\text{MAP}} \cap \mathbf{y}^*|}{|\mathbf{y}^{\text{MAP}}|}, \quad \text{Recall} = \frac{|\mathbf{y}^{\text{MAP}} \cap \mathbf{y}^*|}{|\mathbf{y}^*|}. \quad (15)$$

All three quantities are between 0 and 1, and higher values are better.

#### 5.1.2 MAP Inference

We conduct the MAP inference of DPP by brute-force search on the synthetic data, and turn to the so called minimum Bayes risk (MBR) decoding [Goel and Byrne, 2000, Kulesza and Taskar, 2012] for larger ground sets on real data.

The MBR inference samples subsets  $\mathcal{S} = \{\mathbf{y}_1, \dots, \mathbf{y}_T\}$  from the learned DPP and outputs the one  $\hat{\mathbf{y}}$  which achieves

the highest consensus with the others, where the consensus can be measured by different evaluation metrics depending on applications. We use the F-score in our case. Particularly,

$$\hat{\mathbf{y}} \leftarrow \arg \max_{\mathbf{y}_{t'} \in \mathcal{S}} \frac{1}{T} \sum_{t=1}^T \text{F-SCORE}(\mathbf{y}_{t'}, \mathbf{y}_t). \quad (16)$$

Note that the MBR inference has actually introduced some degrees of flexibility to DPP (and to other probabilistic models). It allows users to infer the desired output according to different evaluation metrics. As a result, the selected subset is not necessarily the “true” diverse subset, but is biased towards the users’ specific interests.

## 5.2 SYNTHETIC DATASET

### 5.2.1 Data

Our ground set has 10 items,  $\mathcal{Y} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{10}\}$ . For each item, we sample a 5-dimensional feature vector from a spherical Gaussian:  $\mathbf{x}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ . To generate the  $\mathbf{L}$  matrix for the DPP, we follow the model in eq. (7); for the parameter vector  $\boldsymbol{\theta}$  we sample from a spherical Gaussian,  $\boldsymbol{\theta} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ , and for the similarity we simply let  $\phi_i = \mathbf{x}_i$  and compute  $S_{ij} = \phi_i^T \phi_j$ .

We identify the most diverse subset  $\mathbf{y}^*$  (eq. (5)) via exhaustive search of all subsets, which is possible given the small ground set. The resulting  $\mathbf{y}^*$  has 5 items on average. We then add noise by randomly (with probability 0.1) adding or dropping an item to or from  $\mathbf{y}^*$ . We repeat the process of sampling another pair of the ground set and its most diverse set. We do so 200 times and use 100 pairs for holdout and 100 for testing. We repeat the process to yield training sets of various sizes.

### 5.2.2 Learning

We compare our large-margin approach using the Hamming loss (eq. (10)) to the standard MLE method for learning DPP parameters.<sup>1</sup> All hyperparameters are tuned by cross-validation. After learning, we apply MAP inference to the testing ground sets.

### 5.2.3 Results

The DPP is parameterized by two things:  $\boldsymbol{\theta}$  for the quality of the items, and  $S_{ij}$  for the similarity among them. Since the ground-truth parameters are known to us, we conduct experiments to isolate the impact of learning either one.

Fig. 1(a) contrasts the two methods when learning  $\boldsymbol{\theta}$  only, assuming all  $S_{ij}$  are known and the ground-truths

<sup>1</sup>Adding a zero-mean Gaussian prior over  $\boldsymbol{\theta}$  while learning with MLE, as in [Kulesza and Taskar, 2011b], did not yield improvement.

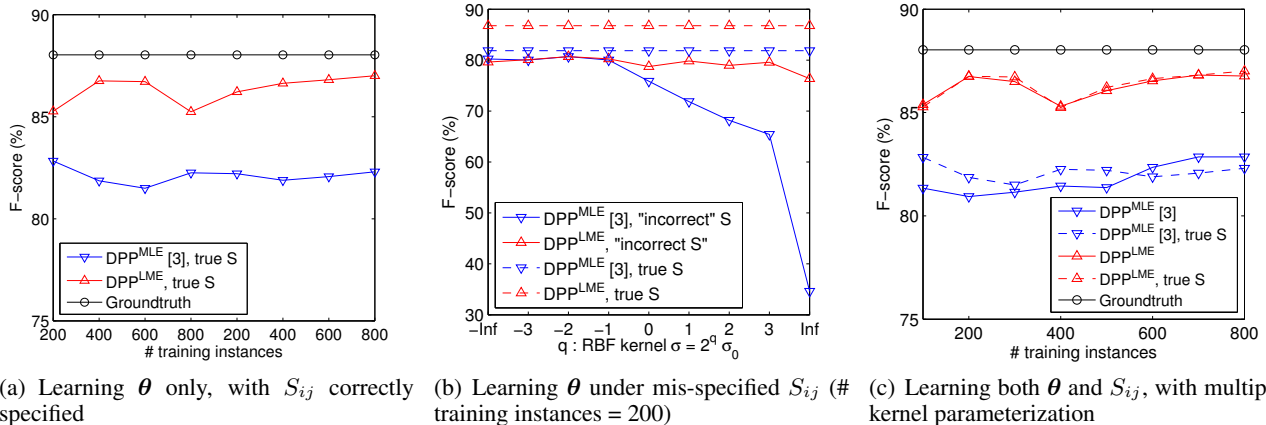


Figure 1: On synthetic datasets, our method DPP<sup>LME</sup> significantly outperforms the state-of-the-art parameter estimation technique DPP<sup>MLE</sup> [Kulesza and Taskar, 2011b] in various learning settings. See text for details. Best viewed in color.

are used. Our DPP<sup>LME</sup> method significantly outperforms DPP<sup>MLE</sup>. When the number of training samples is increased, the performance of our method generally improves and gets very close to the oracle’s performance, for which the true values of *both*  $S_{ij}$  and  $\theta$  are used.

Fig. 1(b) examines the two methods in the setting of model mis-specification, where the  $S_{ij}$  values deliberately deviate from the true values. Specifically, we set them to  $\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2 / \sigma^2)$  where the bandwidth  $\sigma$  varies from small to large, while the true values are  $\mathbf{x}_i^T \mathbf{x}_j$ . All methods generally suffer. However, our method is fairly robust to the mis-specification while DPP<sup>MLE</sup> quickly deteriorates. Our advantage is likely due to our method’s focus on learning to reduce subset selection errors, whereas MLE focuses on learning the right probabilistic model (even if it is already mis-specified).

Fig. 1(c) compares the two methods when both  $\theta$  and  $S_{ij}$  need to be learned from the data. We apply our multiple kernel parameterization technique to model  $S_{ij}$ , as in eq. (8), except  $\beta$  is set to be zero to avoid including the ground-truth. We see that our parameterization overcomes the problems of model mis-specification in Fig. 1(b), demonstrating its effectiveness in approximating unknown similarities. In fact, both learning methods match the performance of the corresponding methods with ground-truth similarity values, respectively. Nonetheless, our large-margin estimation still outperforms MLE significantly.

In summary, our results on synthetic data are very encouraging. Our multiple kernel parameterization avoids the pitfall of model mis-specification, and the large-margin estimation outperforms MLE due to its ability to track selection errors more closely.

### 5.3 DOCUMENT SUMMARIZATION

Next we apply DPP to the task of extractive multi-document summarization [Dang, 2005, Kulesza and

Taskar, 2011b, Lin and Bilmes, 2010]. In this task, the input is a document cluster consisting of several documents on a single topic. The desired output is a subset of the sentences in the cluster that serve as a summary for the entire cluster. Naturally, we want the sentences in this subset to be both representative and diverse.

#### 5.3.1 Experimental Setting

We use the text data from Document Understanding Conference (DUC) 2003 and 2004 [Dang, 2005] as the training and testing sets, respectively. There are 60 document clusters in DUC 2003 and 50 in DUC 2004, each collected over a short time period on a single topic. A cluster includes 10 news articles and on average 250 sentences. Four human reference summaries are provided along with each cluster. Following prior work, we generate the oracle/ground-truth summary by identifying a subset of the original sentences that best agree with the human reference summaries [Kulesza and Taskar, 2011b]. On average, the oracle summary consists of 5 sentences. As is standard practice, we use the oracles only during training. During testing, the algorithm output is evaluated against each of the four human reference summaries separately, and we report the average accuracy [Dang, 2005, Kulesza and Taskar, 2011b, Lin and Bilmes, 2010].

We use the widely-used evaluation package ROUGE [Lin, 2004], which scores document summaries based on  $n$ -gram overlap statistics. We use ROUGE 1.5.5 along with WordNet 2.0, and report the F-score (F), Precision (P), and Recall (R) of both unigram and bigram matchings, denoted by ROUGE-1X and ROUGE-2X respectively ( $X \in \{F, P, R\}$ ). Additionally, we limit the maximum length of each summary to be 665 characters to be consistent with existing work [Dang, 2005]. This yields 5 sentences on average for subsets generated by our algorithm.

To allow the fairest comparison to existing DPP work for

Table 1: Accuracy on document summarization. Our methods outperform others with statistical significance.

| Method  | ROUGE-1F          | ROUGE-1P          | ROUGE-1R          | ROUGE-2F         | ROUGE-2P         | ROUGE-2R         |
|---|-------------------|-------------------|-------------------|------------------|------------------|------------------|
| PEER 35 [Dang, 2005]                                | 37.54             | 37.69             | 37.45             | 8.37             | –                | –                |
| PEER 104 [Dang, 2005]                               | 37.12             | 36.79             | 37.48             | 8.49             | –                | –                |
| PEER 65 [Dang, 2005]                                | 37.87             | 37.58             | 38.20             | 9.13             | –                | –                |
| DPP <sup>MLE</sup> +COS [Kulesza and Taskar, 2011b] | 37.89±0.08        | 37.37±0.08        | 38.46±0.08        | 7.72±0.06        | 7.63±0.06        | 7.83±0.06        |
| Ours (DPP <sup>LME</sup> +COS)                      | 38.36±0.09        | 37.72±0.10        | 39.07±0.08        | 8.20±0.07        | 8.07±0.07        | 8.35±0.07        |
| Ours (DPP <sup>MLE</sup> +MKR)                      | 39.14±0.08        | 39.03±0.09        | 39.31±0.09        | 9.25±0.08        | 9.24±0.08        | 9.27±0.08        |
| Ours (DPP <sup>LME</sup> +MKR)                      | <b>39.71±0.05</b> | <b>39.61±0.08</b> | <b>39.87±0.06</b> | <b>9.40±0.08</b> | <b>9.38±0.08</b> | <b>9.43±0.08</b> |

this task, we use the same features designated in [Kulesza and Taskar, 2011b]. To model quality, the features are the sentence length, position in the original document, mean cluster similarity, LexRank [Erkan and Radev, 2004], and personal pronouns. To model the similarity, the features are the standard normalized term frequency-inverse document frequency (tf-idf) vectors.

### 5.3.2 Learning

We consider two ways of modeling similarities. The first one is to use the cosine similarity (COS) between feature vectors, as in [Kulesza and Taskar, 2011b]. The second is our multiple kernel based similarity (MKR, eq. (8)). For MKR, the bandwidths are  $\sigma = 2^q$ ,  $q = -6, -5, \dots, 6$ , and the combination coefficients are learned on the data. We implement the method in [Kulesza and Taskar, 2011b] as a baseline (DPP<sup>MLE</sup>+COS). We also test an enhanced variant of that method by replacing its cosine similarity with our multiple kernel based similarity (DPP<sup>MLE</sup>+MKR).

### 5.3.3 Results

Table 1 compares several DPP-based methods, as well as the top three results (PEER 35, 104, 65) from the DUC 2004 competition, which are not DPP-based (“–” indicates results not available). Since the DPP MAP inference is NP-hard, we use a sampling technique to extract the most diverse subset [Kulesza and Taskar, 2012]. We run inference 10 times and report the mean accuracy and standard error.

The state-of-the-art MLE-trained DPP model (DPP<sup>MLE</sup>+COS) [Kulesza and Taskar, 2011b] achieves about the same performance as the best PEER results of DUC 2004. We obtain a noticeable improvement by applying our large-margin estimation (DPP<sup>LME</sup>+COS). By applying multiple kernels to model similarity, we obtain significant improvements (above the standard errors) for both parameter estimation techniques. In particular, our complete method, DPP<sup>LME</sup>+MKR, attains the best performance across all the evaluation metrics.

## 5.4 VIDEO SUMMARIZATION

Finally, we demonstrate the broad applicability of our method by applying it to video summarization. In this case, the goal is to select a set of representative and diverse

frames from a video sequence.

### 5.4.1 Experimental Setting

The dataset consists of 50 videos from the Open Video Project (OVP)<sup>2</sup>. They are 30fps, 352×240 pixels, vary from 1 to 4 minutes, and are distributed across several genres including documentary, educational, historical, etc. We use the provided ground truth key frame summaries [de Avila et al., 2011], where each video is labeled by five annotators independently. We perform 5-fold validation and report the average result. We apply several preprocessing steps to remove frames that are trivially redundant (due to high temporal correlation) or of low visual quality. We use a similar procedure as in the document summarization task to generate the oracle/ground-truth subsets. On average, the ground-truth has 9 frames (in contrast, our method yields subsets from 5 to 20 frames). We use the public evaluation package VSUMM to evaluate the system-generated summary frames and again compute Precision, Recall and F-score [de Avila et al., 2011]. More details are in the supplementary material.

### 5.4.2 Features

We extract from each frame a color histogram and SIFT-based Fisher vector [Lowe, 2004, Perronnin and Dance, 2007] to model pairwise frame similarity  $S_{ij}$ . The two features are combined via our multiple kernel representation. To model the quality of each frame, we extract both intra-frame and inter-frame representativeness features. They are computed on the saliency maps [Rahtu et al., 2010, Hou et al., 2012] and include the mean, standard deviation, median, and quantiles of the maps as well as the the visual similarities between a frame and its neighbors. We z-score them within each video sequence.

### 5.4.3 Results

Table 2 compares several methods for selecting key frames: an unsupervised clustering method VSUMM [de Avila et al., 2011] (we implemented its two variants, offering a degree of tradeoff between precision and recall, and finely tuned the parameters), DPP<sup>MLE</sup> with a multiple kernel parameterization of  $S_{ij}$ , and our margin-based approach. For

<sup>2</sup>The Open Video Project: [www.open-video.org](http://www.open-video.org)

Table 2: Accuracy on video summarization. Our method performs the best and allows precision-recall control.

| Metric    | VSUMM1                  | VSUMM2                  | DPP <sup>MLE</sup> +MKR | Ours (DPP <sup>MLE</sup> +MKR) |                   |                   |
|-----------|-------------------------|-------------------------|-------------------------|--------------------------------|-------------------|-------------------|
|           | [de Avila et al., 2011] | [de Avila et al., 2011] |                         | $\omega = 1/64$                | $\omega = 1$      | $\omega = 64$     |
| F-score   | 70.25                   | 68.20                   | 72.94±0.08              | 71.25±0.09                     | <b>73.46±0.07</b> | 72.39±0.10        |
| Precision | 70.57                   | 73.14                   | 68.40±0.08              | <b>74.00±0.09</b>              | 69.68±0.08        | 67.19±0.11        |
| Recall    | 75.77                   | 69.14                   | 82.51±0.11              | 72.71±0.11                     | 81.39±0.09        | <b>83.24±0.09</b> |

our method, we illustrate its flexibility to target different operating points, by varying the tradeoff constant  $\omega$  in the generalized Hamming distance loss function eq. (11). Recall that higher values of  $\omega$  will promote higher recall, while lower promote higher precision.

The results clearly demonstrate the advantage of our approach, particularly in how it offers finer control of the tradeoff between precision and recall. By adjusting  $\omega$ , our method performs the best in each of the three metrics and outperforms the baselines by a statistically significant margin measured in the standard errors. Controlling the tradeoff is quite valuable in this application; for example, high precision may be preferable to a user summarizing a video he himself captured (he knows what appeared in the video, and wants a noise-free summary), whereas high recall may be preferable to a user summarizing a video taken by a third party (he has not seen the original video, and prefers some noise to dropped frames).

More details are illustrated in Fig. 2, in which by varying  $\omega$  from  $2^{-6}$  to  $2^8$  we obtain 8 pairs of (precision, recall) values. We apply uniform interpolation among them and draw the precision-recall curve. One can see that DPP<sup>MLE</sup> is able to control the characteristics of the DPP generated summaries, biasing them to either high precision or high recall and without sacrificing the other too much. Though MLE or VSUMM does not supply such modeling flexibility, we also include them in the figure for reference.

We also present qualitative results on video summarization in Fig. 3. For this particular video, DPP<sup>MLE</sup>, DPP<sup>MLE</sup> with  $\omega = 1$ , and DPP<sup>MLE</sup> with  $\omega = 64$  all give rise to high recalls. Their output summaries are pretty lengthy, and may be boring to some users who just want to grasp something interesting to watch. By turning down the weight to  $\omega = 1/64$ , our DPP<sup>MLE</sup> dramatically improves the precision to 76% (in contrast to the 48% of DPP<sup>MLE</sup>).

### 5.5 COMPUTATIONAL COMPLEXITY

The computational complexities of MLE and our large-margin approach are the same,  $\mathcal{O}(N \times D^3)$ , for computing the objective functions. Here  $N$  is the number of training instances, and  $D$  is the size of the largest ground set. Our softmax trick allows us to handle tractably an exponentially large number of constraints. Using gradient descent in parameter learning, the computational complexity in each iteration is thus also  $\mathcal{O}(N \times D^3)$  for both methods. MLE is slightly faster (20% measured in wall-clock time).

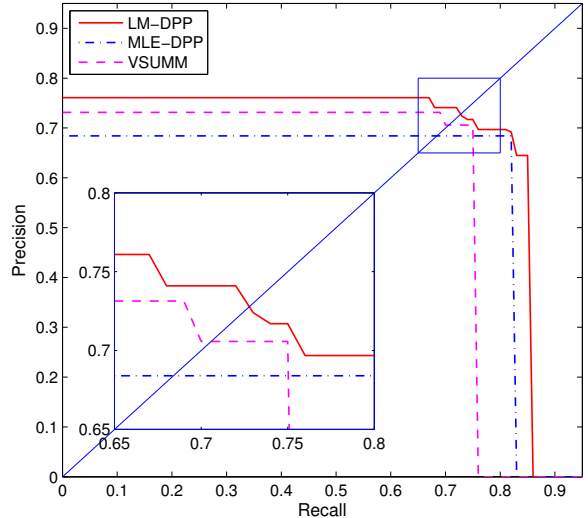


Figure 2: Balancing precision and recall. Through our large-margin DPPs (DPP<sup>MLE</sup>), we can balance precision and recall by varying  $\omega$  in the generalized Hamming distance. In contrast, neither MLE nor VSUMM (the two variants in [de Avila et al., 2011]) are plotted together) is readily able to support such flexibility.

## 6 CONCLUSION

The determinantal point process (DPP) offers a powerful and probabilistically grounded approach for selecting diverse subsets. We proposed a novel technique for learning DPPs from annotated data. In contrast to the status quo of maximum likelihood estimation, our method is more flexible in modeling pairwise similarity and avoids the pitfall of model mis-specification. Empirical results demonstrate its advantages on both synthetic datasets and challenging real-world summarization applications.

### Acknowledgements

F. S., W. C., and B. G. are supported by ARO Award #W911NF-12-1-0241, DARPA Award #D11AP00278, NSF IIS Award #1065243, ONR #N000141210066, and Alfred P. Sloan Fellowship. K. G. is supported by ONR YIP #N00014-12-1-0754.

### References

R. H. Affandi, A. Kulesza, and E. B. Fox. Markov determinantal point processes. In *UAI*, 2012.





Figure 3: Video summaries generated by  $\text{DPP}^{\text{MLE}}$  and our  $\text{DPP}^{\text{LME}}$  with  $\omega = 1$ ,  $\omega = 2^6$ , and  $\omega = 2^{-6}$ , respectively. The oracle summary is also included for reference.

- R. H. Affandi, E. B. Fox, and B. Taskar. Approximate inference in continuous determinantal point processes. In *NIPS*, 2013.
- R. H. Affandi, E. B. Fox, R. P. Adams, and B. Taskar. Learning the parameters of determinantal point process kernels. In *ICML*, 2014.
- R. Burton and R. Pemantle. Local characteristics, entropy and limit theorems for spanning trees and domino tilings via transfer-impedances. *The Annals of Probability*, pages 1329–1371, 1993.
- H. T. Dang. Overview of duc 2005. In *Document Understanding Conf.*, 2005.
- S. E. F. de Avila, A. P. B. Lopes, et al. Vsumm: A mechanism designed to produce static video summaries and a novel evaluation method. *Pattern Recognition Letters*, 32(1):56–68, 2011.
- G. Erkan and D. R. Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *JAIR*, 22(1):457–479, 2004.
- J. Gillenwater, A. Kulesza, and B. Taskar. Near-optimal map inference for determinantal point processes. In *NIPS*, 2012.
- V. Goel and W. J. Byrne. Minimum bayes-risk automatic speech recognition. *Computer Speech & Language*, 14(2):115–135, 2000.
- X. Hou, J. Harel, and C. Koch. Image signature: High-lighting sparse salient regions. *T-PAMI*, 34(1):194–201, 2012.
- T. Jebara. *Machine learning: discriminative and generative*. Springer, 2004.
- B. Kang. Fast determinantal point process sampling with application to clustering. In *NIPS*, 2013.
- C.-W. Ko, J. Lee, and M. Queyranne. An exact algorithm for maximum entropy sampling. *Operations Research*, 43(4):684–691, 1995.
- A. Kulesza and B. Taskar. k-dpps: Fixed-size determinantal point processes. In *ICML*, 2011a.
- A. Kulesza and B. Taskar. Learning determinantal point processes. In *UAI*, 2011b.
- A. Kulesza and B. Taskar. Structured determinantal point processes. In *NIPS*, 2011c.
- A. Kulesza and B. Taskar. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*, 5(2-3):123–286, 2012.
- C.-Y. Lin. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proc. of the ACL-04 Workshop*, 2004.
- H. Lin and J. Bilmes. Multi-document summarization via budgeted maximization of submodular functions. In *NAACL/HLT*, 2010.
- D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.
- O. Macchi. The coincidence approach to stochastic point processes. *Advances in Applied Probability*, 7(1):83–122, 1975.
- A. Y. Ng and M. I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *NIPS*, 2002.
- F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In *CVPR*, 2007.

- E. Rahtu, J. Kannala, M. Salo, and J. Heikkil. Segmenting salient objects from images and videos. In *ECCV*, 2010.
- F. Sha and L. K. Saul. Large margin hidden markov models for automatic speech recognition. In *NIPS*, 2006.
- B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In *NIPS*, 2004.
- B. Taskar, V. Chatalbashev, D. Koller, and C. Guestrin. Learning structured prediction models: A large margin approach. In *ICML*, 2005.
- I. Tsochantaris, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *ICML*, 2004.
- V. Vapnik. *Statistical learning theory*. 1998. Wiley, New York, 1998.

---

# Fast Relative-Error Approximation Algorithm for Ridge Regression

---

Shouyuan Chen<sup>1\*</sup> Yang Liu<sup>21\*</sup> Michael R. Lyu<sup>31</sup> Irwin King<sup>31</sup> Shengyu Zhang<sup>21</sup>

3: Shenzhen Key Laboratory of Rich Media Big Data Analytics and Applications,  
Shenzhen Research Institute, The Chinese University of Hong Kong, Shenzhen, China

2: The Institute of Theoretical Computer Science and Communications, The Chinese University of Hong Kong

1: Department of Computer Science and Engineering, The Chinese University of Hong Kong  
chenshouyuan@gmail.com {yliu,lyu,king,syzhang}@cse.cuhk.edu.hk

## Abstract

Ridge regression is one of the most popular and effective regularized regression methods, and one case of particular interest is that the number of features  $p$  is much larger than the number of samples  $n$ , i.e.  $p \gg n$ . In this case, the standard optimization algorithm for ridge regression computes the optimal solution  $\mathbf{x}^*$  in  $O(n^2p + n^3)$  time. In this paper, we propose a fast *relative-error* approximation algorithm for ridge regression. More specifically, our algorithm outputs a solution  $\tilde{\mathbf{x}}$  satisfying  $\|\tilde{\mathbf{x}} - \mathbf{x}^*\|_2 \leq \epsilon \|\mathbf{x}^*\|_2$  with high probability and runs in  $\tilde{O}(\text{nnz}(\mathbf{A}) + n^3/\epsilon^2)$  time, where  $\text{nnz}(\mathbf{A})$  is the number of non-zero entries of matrix  $\mathbf{A}$ .

To the best of our knowledge, this is the first algorithm for ridge regression that runs in  $o(n^2p)$  time with provable relative-error approximation bound on the output vector. In addition, we analyze the risk inflation bound of our algorithm and apply our techniques to two generalizations of ridge regression, including multiple response ridge regression and a non-linear ridge regression problem. Finally, we show empirical results on both synthetic and real datasets.

## 1 INTRODUCTION

Ridge regression is one of the most popular and effective regularized regression methods, and one case of particular interest is that the number of features  $p$  is much larger than the number of samples  $n$ , i.e.  $p \gg n$ . The definition of ridge regression problem is as follows. Given an  $n \times p$  sample-by-feature design matrix  $\mathbf{A}$  together with an  $n$ -dimensional target vector  $\mathbf{b}$ , and a parameter  $\lambda > 0$ , the goal of ridge regression is to find a  $p$ -dimensional vector

$\mathbf{x}^*$  such that

$$\mathbf{x}^* = \arg \min_{\mathbf{x} \in \mathbb{R}^p} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{x}\|_2^2. \quad (1)$$

Saunders et al. [29] showed that that the unique minimizer of Eq. (1) can be computed as follows

$$\mathbf{x}^* = \mathbf{A}^T(\mathbf{A}\mathbf{A}^T + \lambda\mathbf{I}_n)^{-1}\mathbf{b}. \quad (2)$$

Using Eq. (2), the time complexity of computing  $\mathbf{x}^*$  is  $O(n^2p + n^3)$ , which is  $O(n^2p)$  when  $p \gg n$ . This approach is widely applied in practice for the  $p > n$  cases. However, for large dataset with high dimensional features, i.e.  $p \gg n \gg 1$ , this approach can be prohibitively slow.

**Our contributions.** In this paper, we present the first algorithm for ridge regression that runs in  $o(n^2p)$  time with a provable relative-error approximation bound on the output vector. Specifically, our proposed algorithm outputs a vector  $\tilde{\mathbf{x}}$  such that  $\|\tilde{\mathbf{x}} - \mathbf{x}^*\|_2 \leq \epsilon \|\mathbf{x}^*\|_2$  with high probability without any assumptions on input  $\mathbf{A}$  and  $\mathbf{b}$ . We show that our proposed algorithm runs in  $O(\text{nnz}(\mathbf{A}) + n^2r \log(\frac{r}{\epsilon})/\epsilon^2)$  time, where  $\text{nnz}(\mathbf{A})$  is the number of non-zero entries of  $\mathbf{A}$  and  $r$  is the rank of  $\mathbf{A}$ . Since  $\text{nnz}(\mathbf{A}) \leq np \ll n^2p$  and  $r \leq n$ , our algorithm is substantially faster than the existing approach, which uses  $O(n^2p + n^3)$  for  $p \gg n$  instances, even if  $\mathbf{A}$  is full rank (in this case, our algorithm runs in  $O(\text{nnz}(\mathbf{A}) + n^3 \log(\frac{n}{\epsilon})/\epsilon^2)$  time).

For supplements to our main result, we also prove a risk inflation bound of our algorithm under standard statistical assumptions. In addition, we apply our techniques to several generalizations of ridge regression. In particular, we extend our algorithm to multiple ridge regression problem, where there are multiple target vectors. Similar to our result for standard ridge regression, we also prove a relative-error bound on the output. Moreover, building upon the recent results [3], we extend our techniques and design a fast relative-error approximation algorithm for a non-linear ridge regression problem.

We evaluate our algorithm on both synthetic and real datasets. The experimental results support our theoretical

---

The first two authors contributed equally.

analysis and demonstrate that the proposed algorithm can achieve considerable speed gains and produce highly accurate outputs.

### 1.1 Related work

**Oblivious subspace embedding (OSE).** Sarlós [28] pioneered the use of OSEs for accelerating approximation algorithms for numerical linear algebra problems, including matrix multiplication, linear regression and low rank approximation. His algorithms were later improved by several work [12, 24] using different OSEs such as sparse embedding. More recent work showed that OSEs can be used to speed up other problems as well, including  $k$ -means clustering [10], approximating leverage scores [23], canonical correlation analysis [2], support vector machine [25], SVD-truncated linear regression [9], and non-linear regression [3]. In this line of research, approximation algorithms of linear regression [28, 12, 24, 27] are the most relevant to ours. However, all of these work focused on  $n \gg p$  instances and the fastest algorithm [24] among them runs in  $O(\text{nnz}(\mathbf{A}) + p^3 \log(p)/\epsilon^2)$  time, which is inefficient in  $p \gg n$  cases. Moreover, they used the sketched matrix  $\mathbf{SA}$  which reduces the sample size  $n$  while we use  $\mathbf{AS}^T$  which reduces feature dimension  $p$ . This distinction leads to a very different design and analysis of algorithms.

**Ridge regression.** The bottlenecks of solving ridge regression are constructing and inverting the kernel matrix  $\mathbf{AA}^T$ . In the mean time, fast kernel approximation algorithms for large datasets have been a research focus for many years. Many approximation schemes are highly successful such as low rank approximation [33], Nyström methods [7, 15, 19, 34], sampling [20, 1], incomplete Cholesky factorization [5, 16] and specialized techniques for certain classes of kernels [26, 21]. We notice that many of these proposals focused on speeding up the  $n \gg p$  cases but did not necessarily improve the  $p \gg n$  cases. More importantly, from these work, it is not clear how the error of kernel approximation impacts on the accuracy of the approximation result of ridge regression. This problem was recently studied in several work [13, 4, 35] under different settings. However, none of these work provided a relative-error approximation guarantee on the output vector.

The work most closely related to our results is [22]. They proposed an approximation algorithm for ridge regression by accelerating the computation of kernel matrix  $\mathbf{AA}^T$  using subsampled randomized Hadamard transformation (SRHT). Their algorithm runs in  $O(np \log(n)/\epsilon^2 + n^3)$  time, which is  $o(n^2p)$  time. However, their algorithm does not have a provable guarantee on the error of the output vector. In addition, the risk inflation bound they proved might not hold since their proof is based on a problematic claim. We have included a detailed argument and counterexample for their proof in our supplementary material (see Section E).

## 2 PRELIMINARIES

### 2.1 NOTATION

Let  $[k]$  denote the set of integers  $\{1, 2, \dots, k\}$ . Given a matrix  $\mathbf{M} \in \mathbb{R}^{n \times p}$  of rank  $r$ . For  $i \in [n]$ , let  $\mathbf{M}_{(i)}$  denote the  $i$ -th row of  $\mathbf{M}$  as a row vector. Let  $\text{nnz}(\mathbf{M})$  denote the number of non-zero entries of  $\mathbf{M}$ . Let  $\|\mathbf{M}\|_F$  denote the Frobenius norm of  $\mathbf{M}$  and let  $\|\mathbf{M}\|_2$  denote the spectral norm of  $\mathbf{M}$ . Let  $\sigma_i(\mathbf{M})$  denote the  $i$ -th largest singular value of  $\mathbf{M}$  and let  $\sigma_{\max}(\mathbf{M})$  and  $\sigma_{\min}(\mathbf{M})$  denote the largest and smallest singular values of  $\mathbf{M}$ . The thin SVD of  $\mathbf{M}$  is  $\mathbf{M} = \mathbf{U}_M \mathbf{\Sigma}_M \mathbf{V}_M^T$ , where  $\mathbf{U}_M \in \mathbb{R}^{n \times r}$ ,  $\mathbf{\Sigma}_M \in \mathbb{R}^{r \times r}$  and  $\mathbf{V}_M \in \mathbb{R}^{p \times r}$ .

The Moore-Penrose pseudoinverse of  $\mathbf{M}$  is a  $p \times n$  matrix defined by  $\mathbf{M}^\dagger = \mathbf{V}_M \mathbf{\Sigma}_M^{-1} \mathbf{U}_M^T$ , which can be computed in  $O(n^2p)$  time when  $p > n$ . Finally, let  $\mathbf{I}_n$  denote the  $n \times n$  identity matrix and let  $\mathbf{0}_n$  denote the  $n \times n$  zero matrix.

### 2.2 OBLIVIOUS SUBSPACE EMBEDDING

We start by reviewing the definition of oblivious subspace embedding (OSE).

**Definition 1.** *Given any  $r > 0$ ,  $\delta \in (0, 1)$  and  $\epsilon \in (0, 1)$ , we call a  $t \times p$  random matrix  $\mathbf{S}$  an  $(r, \delta, \epsilon)$ -OSE, if, for any rank  $r$  matrix  $\mathbf{M} \in \mathbb{R}^{p \times m}$ , the following holds simultaneously for all  $\mathbf{z} \in \mathbb{R}^m$ ,*

$$(1 - \epsilon) \|\mathbf{Mz}\|_2 \leq \|\mathbf{SMz}\|_2 \leq (1 + \epsilon) \|\mathbf{Mz}\|_2,$$

with probability at least  $1 - \delta$ .

In fact, many random matrices, which are widely used in machine learning, have been shown to be OSEs, e.g. Gaussian matrices [14] and random sign matrices [28]. However, many of these matrices are dense. For a dense OSE  $\mathbf{S} \in \mathbb{R}^{t \times p}$ , computing sketched matrix  $\mathbf{SM}$  given  $\mathbf{M} \in \mathbb{R}^{p \times m}$  requires time  $O(t \cdot \text{nnz}(\mathbf{M}))$ . To speed up the computation of the sketched matrix  $\mathbf{SM}$ , several work sought  $\mathbf{S}$  supporting fast matrix-vector multiplication [12, 32, 24]. We refer interested readers to [8] for an overview of the development of fast OSEs.

In this paper, we use a combination of two types of fast OSEs: *sparse embedding* and *subsampled randomized Hadamard transformation* (SRHT). In the following, we review their definitions and key properties.

**Sparse embedding.** A sparse embedding matrix  $\mathbf{\Phi}_{\text{sparse}}$  is a very sparse matrix such that there is only one non-zero element per column.  $\mathbf{\Phi}_{\text{sparse}} \in \mathbb{R}^{t \times p}$  can be constructed as follows. Let  $h : [p] \rightarrow [t]$  be a random mapping such that for each  $i \in [p]$ ,  $h(i)$  is uniformly distributed over  $[t]$ . Let  $\mathbf{\Phi} \in \{0, 1\}^{t \times p}$  be a binary matrix with  $\mathbf{\Phi}_{h(i), i} = 1$  for each  $i \in [p]$  and all remaining entries 0. Let  $\mathbf{D}$  be an  $p \times p$  random diagonal matrix where each diagonal entry is independent chosen  $+1$  or  $-1$  with equal probability. Finally,

the sparse embedding matrix  $\Phi_{\text{sparse}}$  is the product of  $\mathbf{D}$  and  $\Phi$ , i.e.  $\Phi_{\text{sparse}} = \Phi\mathbf{D}$ . It is easy to see that computing  $\Phi_{\text{sparse}}\mathbf{M}$  takes  $O(\text{nnz}(\mathbf{M}))$  time due to the sparsity of  $\Phi_{\text{sparse}}$ .

Recently, Clarkson and Woodruff [12] showed that  $\Phi_{\text{sparse}}$  is an  $(r, \delta, \epsilon)$ -OSE if  $t \geq O(r^2/\epsilon^4)$ . Later, the bound on  $t$  was improved to  $t \geq O(r^2/\epsilon^2)$  by Nelson and Nguyen [24]. Their result is restated in the following.

**Theorem 1.** [24, Theorem 3] *Given  $\epsilon \in (0, 1)$ ,  $\delta \in (0, 1)$  and  $r > 0$ , if  $t \geq \delta^{-1}(r^2 + r)/(2\epsilon - \epsilon^2)^2$ , then sparse embedding matrix  $\Phi_{\text{sparse}} \in \mathbb{R}^{t \times p}$  is an  $(r, \delta, \epsilon)$ -OSE.*

**SRHT.** An SRHT matrix  $\Phi_{\text{srht}}$  is a highly structured matrix which allows fast, FFT-style matrix-vector multiplication. The definition of SRHT matrix  $\Phi_{\text{srht}} \in \mathbb{R}^{t \times p}$  is as follows. Without loss of generality, suppose that  $p$  is a power of 2 (otherwise we can pad a sufficient number of zeros). Then,  $\Phi_{\text{srht}}$  is given by

$$\Phi_{\text{srht}} = \sqrt{\frac{p}{t}} \mathbf{R}\mathbf{H}\mathbf{D},$$

where  $\mathbf{D} \in \mathbb{R}^{p \times p}$  is a random diagonal matrix whose entries are  $+1$  or  $-1$  with equal probability;  $\mathbf{R} \in \mathbb{R}^{t \times p}$  are  $t$  rows from the  $p \times p$  identity matrix, where the rows are chosen uniformly at random without replacement; and  $\mathbf{H} \in \mathbb{R}^{p \times p}$  is a normalized Walsh-Hadamard matrix, which is defined as

$$\mathbf{H}_k = \begin{bmatrix} \mathbf{H}_{k/2} & \mathbf{H}_{k/2} \\ \mathbf{H}_{k/2} & -\mathbf{H}_{k/2} \end{bmatrix} \text{ with } \mathbf{H}_2 = \begin{bmatrix} +1 & +1 \\ +1 & -1 \end{bmatrix},$$

and  $\mathbf{H} = p^{-\frac{1}{2}} \mathbf{H}_p \in \mathbb{R}^{p \times p}$ .

Using FFT-style algorithms, the product  $\Phi_{\text{srht}}\mathbf{M}$  can be computed in  $O(np \log(p))$  time [32]. Tropp [32] showed that  $\Phi_{\text{srht}}$  is an OSE if  $t \geq O([\sqrt{r} + \sqrt{\log(p)}]^2 \log(r)/\epsilon)$ , or  $t \geq O(r \log(r)/\epsilon)$  when  $r > \log(p)$ . His result is restated in the next theorem.

**Theorem 2.** [32, Lemma 4.1] *Given  $\epsilon \in (0, 1)$ ,  $\delta \in (0, 1)$  and  $r > 0$ , if  $t \geq 6\epsilon^{-1}[\sqrt{r} + \sqrt{8 \log(3p/\delta)}]^2 \log(3r/\delta)$ , then SRHT matrix  $\Phi_{\text{srht}} \in \mathbb{R}^{t \times p}$  is an  $(r, \delta, \epsilon)$ -OSE.*

### 2.3 COMBINATION OF SPARSE EMBEDDING AND SRHT

Sparse embedding matrix is an extremely fast OSE since computing  $\Phi_{\text{sparse}}\mathbf{M}$  takes only  $O(\text{nnz}(\mathbf{M}))$  time, which equals to the complexity of reading  $\mathbf{M}$ . Meanwhile SRHT produces a sketch with  $t = O(r \log(r)/\epsilon)$  rows which is smaller than sparse embedding, which requires  $t = O(r^2/\epsilon^2)$ . In this paper, we use the combination of SRHT and sparse embedding that enjoys the benefits from both of them. Specifically, we consider the product  $\mathbf{S} = \Phi_{\text{srht}}\Phi_{\text{sparse}}$ , where  $\Phi_{\text{srht}}$  is a  $t \times t'$  SRHT matrix and  $\Phi_{\text{sparse}}$  is a  $t' \times p$  sparse embedding matrix. The next theorem shows that, if  $t = O([\sqrt{r} + \sqrt{\log(p)}]^2 \log(r)/\epsilon)$  and  $t' = O(r^2/\epsilon^2)$ , the product  $\mathbf{S}$  is an OSE.

**Theorem 3.** *Given  $\epsilon \in (0, 1)$ ,  $\delta \in (0, 1)$  and  $r > 0$ , select integers  $t' \geq 2\delta^{-1}(r^2 + r)/(2\epsilon/3 - \epsilon^2/9)^2$  and  $t \geq 18\epsilon^{-1}[\sqrt{r} + \sqrt{8 \log(6p/\delta)}]^2 \log(6r/\delta)$ . Let  $\Phi_{\text{sparse}}$  be a  $t' \times p$  sparse embedding matrix and let  $\Phi_{\text{srht}}$  be a  $t \times t'$  SRHT matrix. Then the product  $\mathbf{S} = \Phi_{\text{srht}}\Phi_{\text{sparse}}$  is an  $(r, \delta, \epsilon)$ -OSE.*

Hence, when  $r \geq O(\log(p))$ , the product  $\mathbf{S}$  has  $t = O(r \log(r)/\epsilon)$  rows, which is smaller than only using sparse embedding matrix, and computing a sketched matrix  $\mathbf{S}\mathbf{M}$  given  $\mathbf{M} \in \mathbb{R}^{p \times m}$  takes  $O(\text{nnz}(\mathbf{M}) + mr^2 \log(r)/\epsilon^2)$  time. The proof of Theorem 3 is deferred to the supplementary material.

## 3 ALGORITHMS AND MAIN RESULTS

In this section, we present our approximation algorithm for ridge regression (Algorithm 1). Then, we state our main result on the approximation guarantee of our algorithm (Theorem 4). In Section 3.1, we outline the proofs of our main result.

**Algorithm.** Algorithm 1 takes inputs of the design matrix  $\mathbf{A} \in \mathbb{R}^{n \times p}$ , target vector  $\mathbf{b} \in \mathbb{R}^n$ , regularization parameter  $\lambda > 0$  and integer parameters  $t'$  and  $t$ . The first part of Algorithm 1 is to compute the sketched matrix  $\mathbf{A}\mathbf{S}^T$ , where  $\mathbf{S} \in \mathbb{R}^{t \times p}$  is chosen to be the product of sparse embedding matrix  $\Phi_{\text{sparse}} \in \mathbb{R}^{t' \times p}$  and SRHT matrix  $\Phi_{\text{srht}} \in \mathbb{R}^{t \times t'}$ , i.e.  $\mathbf{S} = \Phi_{\text{srht}}\Phi_{\text{sparse}}$ . As the first step, the algorithm constructs the sparse embedding matrix  $\Phi_{\text{sparse}}$  and the SRHT matrix  $\Phi_{\text{srht}}$ . After that, the algorithm applies  $\Phi_{\text{sparse}}$  and  $\Phi_{\text{srht}}$  to each row  $\mathbf{A}_{(i)}$  and obtains the sketched row  $(\mathbf{A}\mathbf{S}^T)_{(i)} = (\mathbf{A}_{(i)}\Phi_{\text{sparse}}^T)\Phi_{\text{srht}}^T$  for all  $i \in [n]$ . This step can be done in one pass through the rows of  $\mathbf{A}$  in arbitrary order. The algorithm then combines the sketched rows  $\{(\mathbf{A}\mathbf{S}^T)_{(i)}\}_{i \in [n]}$  to form the sketched matrix  $\mathbf{A}\mathbf{S}^T$ .

Next, the algorithm uses the sketched matrix  $\mathbf{A}\mathbf{S}^T$  to compute the approximate solution  $\tilde{\mathbf{x}}$  of ridge regression Eq. (1). In this step, we use the following key estimation of  $\tilde{\mathbf{x}}$ ,

$$\tilde{\mathbf{x}} = \mathbf{A}^T(\mathbf{A}\mathbf{S}^T)^\dagger{}^T(\lambda(\mathbf{A}\mathbf{S}^T)^\dagger{}^T + \mathbf{A}\mathbf{S}^T)^\dagger{} \mathbf{b}. \quad (3)$$

This step requires access to  $\mathbf{A}\mathbf{S}^T$ , which is computed in the previous step, and a second pass through  $\mathbf{A}$  (for pre-multiplying  $\mathbf{A}^T$ ). We summarize the above procedure of computing  $\mathbf{A}\mathbf{S}^T$  and  $\tilde{\mathbf{x}}$  in Algorithm 1.

**Main result.** Our main result is the following theorem which states that, with high probability, the output  $\tilde{\mathbf{x}}$  obtained in Algorithm 1 is a relative-error approximation to the optimal solution  $\mathbf{x}^*$  of ridge regression.

**Theorem 4.** *Suppose that we are given a design matrix  $\mathbf{A} \in \mathbb{R}^{n \times p}$  of rank  $r$ , a target vector  $\mathbf{b} \in \mathbb{R}^n$ , a regularization parameter  $\lambda > 0$ , accuracy parameters  $\epsilon \in (0, 1)$  and  $\delta \in (0, 1)$ . Select integers  $t', t$  such that  $t' \geq 2\delta^{-1}(r^2 + r)/(\epsilon/6 - \epsilon^2/144)^2$  and  $t \geq 72\epsilon^{-1}[\sqrt{r} + \sqrt{8 \log(6p/\delta)}]^2 \log(6r/\delta)$ . Run Algorithm 1 with inputs*

---

**Algorithm 1** Fast relative-error approximation algorithm of ridge regression

---

**Input:** design matrix  $\mathbf{A} \in \mathbb{R}^{n \times p}$  ( $n$  samples with  $p$  features), target vector  $\mathbf{b} \in \mathbb{R}^n$ , regularization parameter  $\lambda > 0$ , integer parameters  $t'$  and  $t$ .

**Output:** approximate solution  $\tilde{\mathbf{x}} \in \mathbb{R}^p$  to ridge regression problem Eq. (1).

- 1: Construct sparse embedding matrix  $\Phi_{\text{sparse}} \in \mathbb{R}^{t' \times p}$ .
  - 2: Construct SRHT matrix  $\Phi_{\text{srht}} \in \mathbb{R}^{t \times t'}$ .
  - 3: **for** each row  $\mathbf{A}_{(i)}$  of  $\mathbf{A}$  in arbitrary order **do**
  - 4:     Compute  $(\mathbf{A}\mathbf{S}^T)_{(i)} \leftarrow (\mathbf{A}_{(i)} \Phi_{\text{sparse}}^T) \Phi_{\text{srht}}^T$
  - 5: **end for**
  - 6: Construct  $\mathbf{A}\mathbf{S}^T$  by concatenating row vectors  $\{(\mathbf{A}\mathbf{S}^T)_{(i)}\}_{i \in [n]}$ .
  - 7: Compute the pseudoinverse  $(\mathbf{A}\mathbf{S}^T)^\dagger$
  - 8: Set  $\tilde{\mathbf{x}} \leftarrow \mathbf{A}^T (\mathbf{A}\mathbf{S}^T)^\dagger^T (\lambda (\mathbf{A}\mathbf{S}^T)^\dagger^T + \mathbf{A}\mathbf{S}^T)^\dagger \mathbf{b}$
  - 9: **return**  $\tilde{\mathbf{x}}$
- 

$\mathbf{A}$ ,  $\mathbf{b}$ ,  $\lambda$ ,  $t'$ ,  $t$  and let  $\tilde{\mathbf{x}}$  denote the output of the algorithm. Then, with probability at least  $1 - \delta$ , we have

$$\|\tilde{\mathbf{x}} - \mathbf{x}^*\|_2 \leq \epsilon \|\mathbf{x}^*\|_2, \quad (4)$$

where  $\mathbf{x}^*$  is the optimal solution of ridge regression in Eq. (1).

In addition, if  $t' = O(r^2/\epsilon^2)$  and  $t = O(r \log(r)/\epsilon)$ , the time complexity of Algorithm 1 is

$$O\left(\text{nnz}(\mathbf{A}) + nr^2 \log\left(\frac{r}{\epsilon}\right) / \epsilon^2 + n^2 r \log(r) / \epsilon\right).$$

**Running times.** Set  $t' = O(r^2/\epsilon^2)$  and  $t = O(r \log(r)/\epsilon)$  according to Theorem 4. Then, the time complexity of each step of Algorithm 1 can be analyzed as follows. Constructing sparse embedding matrix  $\Phi_{\text{sparse}}$  and SRHT matrix  $\Phi_{\text{srht}}$  uses  $O(p)$  time. Right-multiplying the sparse embedding matrix  $\mathbf{A}\Phi_{\text{sparse}}^T$  takes  $O(\text{nnz}(\mathbf{A}))$  time. Computing SRHT  $(\mathbf{A}\Phi_{\text{sparse}}^T)\Phi_{\text{srht}}^T$  uses  $O(nt' \log(t')) = O(nr^2 \log(\frac{r}{\epsilon})/\epsilon^2)$  time. The pseudoinverse of  $\mathbf{A}\mathbf{S}^T$  and  $\lambda(\mathbf{A}\mathbf{S}^T)^\dagger^T + \mathbf{A}\mathbf{S}^T$  can be computed in  $O(n^2 t) = O(n^2 r \log(r)/\epsilon)$  time. Computing the product  $(\mathbf{A}\mathbf{S}^T)^\dagger^T (\lambda(\mathbf{A}\mathbf{S}^T)^\dagger^T + \mathbf{A}\mathbf{S}^T)^\dagger \mathbf{b}$  also takes  $O(n^2 t) = O(n^2 r \log(r)/\epsilon)$  time. Finally, left-multiplying  $\mathbf{A}^T$  uses  $O(\text{nnz}(\mathbf{A}))$  time. So, the total running time is the sum of all these operations, which is  $O(\text{nnz}(\mathbf{A}) + nr^2 \log(\frac{r}{\epsilon})/\epsilon^2 + n^2 r \log(r)/\epsilon)$ .

**Remarks.** In practice, one may not have prior knowledge on the rank  $r$  of  $\mathbf{A}$ . By Theorem 4, it is safe to assume that  $r = n$ , which is the largest possible value of  $r$ , and hence set  $t' = O(n^2/\epsilon^2)$  and  $t = O(n \log(n)/\epsilon)$ . In this case, the running time of Algorithm 1 is  $O(\text{nnz}(\mathbf{A}) + n^3 \log(\frac{n}{\epsilon})/\epsilon^2)$ . This is still an  $o(n^2 p)$  algorithm and is substantially faster than the standard  $O(n^2 p + n^3)$  solver for  $p \gg n$  instances.

In addition, the estimation method of  $\tilde{\mathbf{x}}$  as in Eq. (3) holds for general OSEs  $\mathbf{S}$ , not necessarily limiting to the one used

in Algorithm 1, i.e.  $\mathbf{S} = \Phi_{\text{srht}} \Phi_{\text{sparse}}$ . This fact is formalized in the following lemma.

**Lemma 1.** Given  $\mathbf{A} \in \mathbb{R}^{n \times p}$  of rank  $r$ ,  $\mathbf{b} \in \mathbb{R}^n$ ,  $\lambda > 0$ . Suppose that  $\mathbf{S} \in \mathbb{R}^{t \times p}$  is an  $(r, \delta, \epsilon/4)$ -OSE for  $\epsilon \in (0, 1)$  and  $\delta \in (0, 1)$ . Then, with probability at least  $1 - \delta$ , the approximation solution  $\tilde{\mathbf{x}}$  obtained by Eq. (3) satisfies

$$\|\tilde{\mathbf{x}} - \mathbf{x}^*\|_2 \leq \epsilon \|\mathbf{x}^*\|_2,$$

where  $\mathbf{x}^*$  is the optimal solution to ridge regression Eq. (1).

Our choice of OSE  $\mathbf{S} = \Phi_{\text{srht}} \Phi_{\text{sparse}}$  in Algorithm 1 guarantees that the sketched matrix  $\mathbf{A}\mathbf{S}^T$  can be computed efficiently while has a small number of columns. By Lemma 1, one may use other OSEs as well, for example, SRHT  $\mathbf{S} = \Phi_{\text{srht}}$ . This would lead to a total time complexity of  $O(np \log(p) + n^2 r \log(r)/\epsilon)$ , which is slower than our choice in Algorithm 1 if  $\mathbf{A}$  is a sparse matrix.

### 3.1 PROOF

From this point on, we denote the thin SVD of matrix  $\mathbf{A} \in \mathbb{R}^{n \times p}$  with rank  $r$  by  $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ , with  $\mathbf{U} \in \mathbb{R}^{n \times r}$ ,  $\mathbf{\Sigma} \in \mathbb{R}^{r \times r}$  and  $\mathbf{V} \in \mathbb{R}^{p \times r}$ . We denote the full SVD of  $\mathbf{S}\mathbf{V}$  by  $\mathbf{S}\mathbf{V} = \mathbf{U}_\phi \mathbf{\Sigma}_\phi \mathbf{V}_\phi^T$ , with  $\mathbf{U}_\phi \in \mathbb{R}^{t \times r}$ ,  $\mathbf{\Sigma}_\phi \in \mathbb{R}^{r \times r}$  and  $\mathbf{V}_\phi \in \mathbb{R}^{r \times r}$ . Notice that  $\mathbf{V}_\phi$  is an  $r \times r$  unitary matrix and therefore  $\mathbf{V}_\phi \mathbf{V}_\phi^T = \mathbf{I}_r$ . We will frequently use the following property of the pseudoinverse of matrix product.

**Fact 1.** For any matrix  $\mathbf{A} \in \mathbb{R}^{m \times n}$  and  $\mathbf{B} \in \mathbb{R}^{n \times p}$ , we have  $(\mathbf{A}\mathbf{B})^\dagger = \mathbf{B}^\dagger \mathbf{A}^\dagger$ , if at least one of the following holds.

1.  $\mathbf{A}$  has orthonormal columns.
2.  $\mathbf{B}$  has orthonormal rows.
3.  $\mathbf{A}$  has full column rank and  $\mathbf{B}$  has full row rank.

By Fact 1, we immediately obtain the following lemma.

**Lemma 2.** Suppose that  $\mathbf{S}\mathbf{V}$  is full rank, then the pseudoinverse of  $\mathbf{A}\mathbf{S}^T$  is given by

$$(\mathbf{A}\mathbf{S}^T)^\dagger = (\mathbf{S}\mathbf{V})^\dagger \mathbf{\Sigma}^{-1} \mathbf{U}^T.$$

The first step of our proof is to represent  $\mathbf{x}^*$  and  $\tilde{\mathbf{x}}$  in a form that is easier to work with.

**Lemma 3.** Let the optimal solution of ridge regression  $\mathbf{x}^*$  be defined as in Eq. (2). We have

$$\mathbf{x}^* = \mathbf{V}\mathbf{G}^{-1} \mathbf{U}^T \mathbf{b},$$

where  $\mathbf{G} = \lambda \mathbf{\Sigma}^{-1} + \mathbf{\Sigma}$ .

*Proof.* Consider the full SVD of  $\mathbf{A}$  as  $\mathbf{A} = \mathbf{U}_+ \mathbf{\Sigma}_+ \mathbf{V}_+^T$ , with  $\mathbf{U}_+ \in \mathbb{R}^{n \times n}$ ,  $\mathbf{\Sigma}_+ \in \mathbb{R}^{n \times n}$  and  $\mathbf{V}_+ \in \mathbb{R}^{p \times n}$ . By the relationship between thin SVD and full SVD, we can see that  $\mathbf{U}_+ = [\mathbf{U}, \mathbf{U}_-]$ ,  $\mathbf{\Sigma}_+ = \begin{bmatrix} \mathbf{\Sigma} & \\ & \mathbf{0}_{n-r} \end{bmatrix}$  and  $\mathbf{V}_+ = [\mathbf{V}, \mathbf{V}_-]$ , with  $\mathbf{U}_- \in \mathbb{R}^{n \times (n-r)}$  and  $\mathbf{V}_- \in \mathbb{R}^{p \times (n-r)}$  being column orthonormal matrices.

Now, by definition of  $\mathbf{x}^*$ , we have

$$\begin{aligned}
\mathbf{x}^* &= \mathbf{A}^T(\mathbf{A}\mathbf{A}^T + \lambda\mathbf{I})^{-1}\mathbf{b} \\
&= \mathbf{V}_+\Sigma_+\mathbf{U}_+^T(\mathbf{U}_+\Sigma_+^2\mathbf{U}_+^T + \lambda\mathbf{U}_+\mathbf{U}_+^T)^{-1}\mathbf{b} \\
&= \mathbf{V}_+\Sigma_+(\Sigma_+^2 + \lambda\mathbf{I})^{-1}\mathbf{U}_+^T\mathbf{b} \\
&= \mathbf{V}_+\begin{bmatrix} \Sigma & \\ & \mathbf{0}_{n-r} \end{bmatrix} \begin{bmatrix} (\Sigma^2 + \lambda\mathbf{I}_r)^{-1} & \\ & \lambda^{-1}\mathbf{I}_{n-r} \end{bmatrix} \mathbf{U}_+^T\mathbf{b} \\
&= \mathbf{V}_+\begin{bmatrix} (\Sigma + \lambda\Sigma^{-1})^{-1} & \\ & \mathbf{0}_{n-r} \end{bmatrix} \mathbf{U}_+^T\mathbf{b} \\
&= \mathbf{V}(\Sigma + \lambda\Sigma^{-1})^{-1}\mathbf{U}^T\mathbf{b}.
\end{aligned}$$

□

**Lemma 4.** Define matrix  $\tilde{\mathbf{G}} = \lambda\Sigma^{-1} + \Sigma(\mathbf{S}\mathbf{V})^T(\mathbf{S}\mathbf{V})$ . Let  $\tilde{\mathbf{x}}$  be defined as in Eq. (3). Suppose that  $\mathbf{S}\mathbf{V}$  is full rank. Then, we have that  $\tilde{\mathbf{G}}$  is full rank and that

$$\tilde{\mathbf{x}} = \mathbf{V}\tilde{\mathbf{G}}^{-1}\mathbf{U}^T\mathbf{b}.$$

*Proof.* By the construction of  $\tilde{\mathbf{x}}$ , we have

$$\begin{aligned}
\tilde{\mathbf{x}} &= \mathbf{A}^T((\mathbf{A}\mathbf{S}^T)^\dagger)^T \left( \lambda((\mathbf{A}\mathbf{S}^T)^\dagger)^T + \mathbf{A}\mathbf{S}^T \right)^\dagger \mathbf{b} \\
&= \mathbf{V}\Sigma\mathbf{U}^T\mathbf{U}\Sigma^{-1}(\mathbf{S}\mathbf{V})^\dagger (\lambda\mathbf{U}\Sigma^{-1}(\mathbf{S}\mathbf{V})^\dagger + \mathbf{U}\Sigma(\mathbf{S}\mathbf{V})^T)^\dagger \mathbf{b} \\
&= \mathbf{V}(\mathbf{S}\mathbf{V})^\dagger (\lambda\mathbf{U}\Sigma^{-1}(\mathbf{S}\mathbf{V})^\dagger + \mathbf{U}\Sigma(\mathbf{S}\mathbf{V})^T)^\dagger \mathbf{b} \\
&= \mathbf{V}\mathbf{V}_\phi\Sigma_\phi^{-1}\mathbf{U}_\phi^T \left( \lambda\mathbf{U}\Sigma^{-1}\mathbf{V}_\phi\Sigma_\phi^{-1}\mathbf{U}_\phi^T + \mathbf{U}\Sigma\mathbf{V}_\phi\Sigma_\phi\mathbf{U}_\phi^T \right)^\dagger \mathbf{b} \\
&= \mathbf{V}\mathbf{V}_\phi\Sigma_\phi^{-1}\mathbf{U}_\phi^T\mathbf{U}_\phi \left( \lambda\Sigma^{-1}\mathbf{V}_\phi\Sigma_\phi^{-1} + \Sigma\mathbf{V}_\phi\Sigma_\phi \right)^\dagger \mathbf{U}^T\mathbf{b} \\
&= \mathbf{V}\mathbf{V}_\phi\Sigma_\phi^{-1} \left( \lambda\Sigma^{-1}\mathbf{V}_\phi\Sigma_\phi^{-1} + \Sigma\mathbf{V}_\phi\Sigma_\phi \right)^\dagger \mathbf{U}^T\mathbf{b}, \quad (5)
\end{aligned}$$

where we have repeatedly used Fact 1 and Lemma 2.

Define  $\mathbf{T}_1 = \lambda\Sigma^{-1}\mathbf{V}_\phi\Sigma_\phi^{-1} + \Sigma\mathbf{V}_\phi\Sigma_\phi$ . Next, we show that  $\text{rank}(\mathbf{T}_1) = r$ . To see this, we define  $\mathbf{T}_2 = \lambda\mathbf{I} + \Sigma\mathbf{V}_\phi\Sigma_\phi^2\mathbf{V}_\phi^T\Sigma$  and notice that  $\mathbf{T}_2 = \mathbf{T}_1(\Sigma_\phi\mathbf{V}_\phi^T\Sigma)$ . Since  $\lambda > 0$ , it is clear that  $\mathbf{T}_2$  is a positive definite matrix and therefore  $\text{rank}(\mathbf{T}_2) = r$ .

Now notice that  $\text{rank}(\Sigma_\phi\mathbf{V}_\phi^T\Sigma) = \text{rank}(\Sigma_\phi) = r$ . Hence, we have

$$\text{rank}(\mathbf{T}_1) = \text{rank}(\mathbf{T}_1(\Sigma_\phi\mathbf{V}_\phi^T\Sigma)) = \text{rank}(\mathbf{T}_2) = r.$$

Then, using Fact 1 on  $\Sigma_\phi^\dagger$  and  $\mathbf{T}_1^\dagger$ , we have

$$\begin{aligned}
(5) &= \mathbf{V}\mathbf{V}_\phi \left( \lambda\Sigma^{-1}\mathbf{V}_\phi + \Sigma\mathbf{V}_\phi\Sigma_\phi^2 \right)^\dagger \mathbf{U}^T\mathbf{b} \\
&= \mathbf{V} \left( \lambda\Sigma^{-1} + \Sigma\mathbf{V}_\phi\Sigma_\phi^2\mathbf{V}_\phi^T \right)^\dagger \mathbf{U}^T\mathbf{b} \\
&= \mathbf{V}\tilde{\mathbf{G}}^\dagger\mathbf{U}^T\mathbf{b},
\end{aligned}$$

where we have used Fact 1 again and that  $\mathbf{S}\mathbf{V} = \mathbf{U}_\phi\Sigma_\phi\mathbf{V}_\phi^T$ . Finally, the rank of  $\tilde{\mathbf{G}}$  is given by

$$\text{rank}(\tilde{\mathbf{G}}) = \text{rank}(\mathbf{T}_1\Sigma_\phi\mathbf{V}_\phi^T) = \text{rank}(\mathbf{T}_1) = r.$$

Hence the pseudoinverse of  $\tilde{\mathbf{G}}$  equals to its inverse, i.e.  $\tilde{\mathbf{G}}^\dagger = \tilde{\mathbf{G}}^{-1}$ , and this concludes our proof of the lemma. □

From Lemma 3 and Lemma 4, we see that  $\tilde{\mathbf{x}}$  admits a representation that is very similar to  $\mathbf{x}^*$ . It is clear that the key difference between  $\tilde{\mathbf{x}}$  and  $\mathbf{x}^*$  comes from that of  $\tilde{\mathbf{G}}$  and  $\mathbf{G}$ .

The next lemma (Lemma 5) is our key technical lemma, which shows that  $\tilde{\mathbf{G}}$  is closely related to  $\mathbf{G}$  in the sense that  $\mathbf{G}^{-1}$  is an approximate matrix inversion of  $\tilde{\mathbf{G}}$ .

**Lemma 5.** Given  $\epsilon \in (0, 1/4)$  and  $\delta \in (0, 1)$ . Let  $\mathbf{S}$  be an  $(r, \delta, \epsilon)$ -OSE. Let  $\tilde{\mathbf{G}} = \lambda\Sigma^{-1} + \Sigma(\mathbf{S}\mathbf{V})^T(\mathbf{S}\mathbf{V})$  and  $\mathbf{G} = \lambda\Sigma^{-1} + \Sigma$ . Notice that  $\mathbf{G}$  is invertible and define  $\mathbf{R} = \mathbf{G}^{-1}\tilde{\mathbf{G}} - \mathbf{I}$ . Then, with probability at least  $1 - \delta$ , we have (a)  $\mathbf{S}\mathbf{V}$  is a full rank matrix, (b)  $\|\mathbf{R}\|_2 \leq 2\epsilon + \epsilon^2$ , and (c)

$$\|(\mathbf{I} + \mathbf{R})^{-1}\mathbf{R}\|_2 \leq \frac{2\epsilon + \epsilon^2}{1 - (2\epsilon + \epsilon^2)}.$$

To prove Lemma 5, we need two ingredients from linear algebra and the theory of OSEs. First, we use the following property on the stability of singular values.

**Lemma 6.** [31, Section 1.3.22 (iv)] Let  $\mathbf{C} \in \mathbb{R}^{m \times n}$  and  $\mathbf{D} \in \mathbb{R}^{m \times n}$  be two matrices of the same size. Then, for all  $i \in [\min\{m, n\}]$ ,

$$|\sigma_i(\mathbf{C} + \mathbf{D}) - \sigma_i(\mathbf{C})| \leq \|\mathbf{D}\|_2.$$

Lemma 6 can be regarded as a generalization of Weyl's inequality to singular values of non-Hermitian matrices. We refer readers to [see 31, Section 1.3] for a proof.

The second ingredient we needed is the following characterization of OSEs.

**Theorem 5.** Let  $\mathbf{V} \in \mathbb{R}^{p \times r}$  be a column orthonormal matrix. Let  $\mathbf{S} \in \mathbb{R}^{t \times p}$  be an  $(r, \delta, \epsilon)$ -OSE. Then, with probability  $1 - \delta$  over the choices of  $\mathbf{S}$ , we have that (a)  $\mathbf{S}\mathbf{V}$  is a full rank matrix and (b) for all  $i \in [r]$ , the  $i$ -th largest singular value of  $\mathbf{S}\mathbf{V}$  is bounded by

$$|1 - \sigma_i(\mathbf{S}\mathbf{V})| \leq \epsilon. \quad (6)$$

The proof of Lemma 6 and Theorem 5 is deferred to the supplementary material. Using them, we are now ready to prove Lemma 5.

*Proof of Lemma 5.* Since  $\mathbf{S}$  is an  $(r, \delta, \epsilon)$ -OSE. By Theorem 5, we have that, with probability  $1 - \delta$ ,  $\mathbf{S}\mathbf{V}$  is a full rank matrix and all singular values of  $\mathbf{S}\mathbf{V}$  are bounded in  $[1 - \epsilon, 1 + \epsilon]$ . In the rest of the proof, we assume this holds. And this already proves part (a) of the lemma.

We start with bounding  $\|\mathbf{R}\|_2$ . By the definition of  $\mathbf{R}$ , we have

$$\|\mathbf{R}\|_2 = \left\| \mathbf{G}^{-1}(\tilde{\mathbf{G}} - \mathbf{G}) \right\|_2$$

$$\begin{aligned}
&= \|(\lambda\boldsymbol{\Sigma}^{-1} + \boldsymbol{\Sigma})^{-1}\boldsymbol{\Sigma}((\mathbf{S}\mathbf{V})^T(\mathbf{S}\mathbf{V}) - \mathbf{I})\|_2 \\
&\leq \|(\lambda\boldsymbol{\Sigma}^{-1} + \boldsymbol{\Sigma})^{-1}\boldsymbol{\Sigma}\|_2 \|\mathbf{V}_\phi \boldsymbol{\Sigma}_\phi^2 \mathbf{V}_\phi^T - \mathbf{I}\|_2 \\
&= \|(\lambda\boldsymbol{\Sigma}^{-1} + \boldsymbol{\Sigma})^{-1}\boldsymbol{\Sigma}\|_2 \|\mathbf{V}_\phi \boldsymbol{\Sigma}_\phi^2 \mathbf{V}_\phi^T - \mathbf{V}_\phi \mathbf{V}_\phi^T\|_2 \\
&= \|(\lambda\boldsymbol{\Sigma}^{-1} + \boldsymbol{\Sigma})^{-1}\boldsymbol{\Sigma}\|_2 \|\boldsymbol{\Sigma}_\phi^2 - \mathbf{I}\|_2 \\
&\leq \max_i \frac{\sigma_i}{\lambda\sigma_i^{-1} + \sigma_i} ((1 + \epsilon)^2 - 1) \\
&\leq 2\epsilon + \epsilon^2, \tag{7}
\end{aligned}$$

where we have used the fact that  $\mathbf{V}_\phi$  is a unitary matrix and dropped terms that do not change spectral norm.

Now, we apply Lemma 6 by setting  $\mathbf{C} = \mathbf{I}$  and  $\mathbf{D} = \mathbf{R}$ . Then, for all  $i \in [r]$ , we have

$$\sigma_i(\mathbf{I} + \mathbf{R}) \geq 1 - \|\mathbf{R}\|_2 \geq 1 - (2\epsilon + \epsilon^2). \tag{8}$$

Hence, we have

$$\begin{aligned}
\|(\mathbf{I} + \mathbf{R})^{-1}\mathbf{R}\|_2 &\leq \|(\mathbf{I} + \mathbf{R})^{-1}\|_2 \|\mathbf{R}\|_2 \\
&\leq (\sigma_{\min}(\mathbf{I} + \mathbf{R}))^{-1} \|\mathbf{R}\|_2 \\
&\leq \frac{2\epsilon + \epsilon^2}{1 - (2\epsilon + \epsilon^2)}.
\end{aligned}$$

□

We are now ready to prove our main results: Lemma 1 and Theorem 4.

*Proof of Lemma 1.* Let  $\epsilon' = \epsilon/4$  and recall the definition  $\tilde{\mathbf{R}} = \mathbf{G}^{-1}\tilde{\mathbf{G}} - \mathbf{I}$ . Since  $\mathbf{S}$  is an  $(r, \delta, \epsilon')$ -OSE. By Lemma 5, with probability  $1 - \delta$ , we have that  $\mathbf{S}\mathbf{V}$  is a full rank matrix and that  $\|(\mathbf{I} + \mathbf{R})^{-1}\mathbf{R}\|_2 \leq \frac{2\epsilon' + \epsilon'^2}{1 - (2\epsilon' + \epsilon'^2)}$ . In the rest of the proof, we assume that this event happens.

Since  $\mathbf{S}\mathbf{V}$  is a full rank matrix. Applying Lemma 3 and Lemma 4, we have

$$\begin{aligned}
\|\tilde{\mathbf{x}} - \mathbf{x}^*\|_2 &= \|\mathbf{V}(\tilde{\mathbf{G}}^{-1} - \mathbf{G}^{-1})\mathbf{U}^T\mathbf{b}\|_2 \\
&= \|(\tilde{\mathbf{G}}^{-1} - \mathbf{G}^{-1})\mathbf{U}^T\mathbf{b}\|_2, \tag{9}
\end{aligned}$$

where we have dropped the unitary term  $\mathbf{V}$  which does not change  $l_2$  norm.

Next, we write  $\tilde{\mathbf{G}} = \mathbf{G}(\mathbf{I} + \mathbf{R})$ . This means that  $\tilde{\mathbf{G}}^{-1} = (\mathbf{I} + \mathbf{R})^{-1}\mathbf{G}^{-1}$ . Therefore,

$$\begin{aligned}
(9) &= \|((\mathbf{I} + \mathbf{R})^{-1} - \mathbf{I})\mathbf{G}^{-1}\mathbf{U}^T\mathbf{b}\|_2 \\
&= \|-(\mathbf{I} + \mathbf{R})^{-1}\mathbf{R}\mathbf{G}^{-1}\mathbf{U}^T\mathbf{b}\|_2 \tag{10} \\
&\leq \|(\mathbf{I} + \mathbf{R})^{-1}\mathbf{R}\|_2 \|\mathbf{G}^{-1}\mathbf{U}^T\mathbf{b}\|_2 \\
&= \|(\mathbf{I} + \mathbf{R})^{-1}\mathbf{R}\|_2 \|\mathbf{x}^*\|_2
\end{aligned}$$

$$\begin{aligned}
&\leq \frac{2\epsilon' + \epsilon'^2}{1 - (2\epsilon' + \epsilon'^2)} \|\mathbf{x}^*\|_2 \tag{11} \\
&\leq 4\epsilon' \|\mathbf{x}^*\|_2 = \epsilon \|\mathbf{x}^*\|_2,
\end{aligned}$$

where Eq. (10) follows from matrix inversion lemma, i.e.  $\mathbf{C}^{-1} - \mathbf{D}^{-1} = -\mathbf{C}^{-1}(\mathbf{C} - \mathbf{D})\mathbf{D}^{-1}$  for any squared matrices  $\mathbf{C}$  and  $\mathbf{D}$  of the same size, and Eq. (11) follows from the assumption on  $\|(\mathbf{I} + \mathbf{R})^{-1}\mathbf{R}\|_2$ . □

*Proof of Theorem 4.* It is easy to see that the solution  $\tilde{\mathbf{x}}$  returned by Algorithm 1 is given by Eq. (3) with  $\mathbf{S} = \boldsymbol{\Phi}_{\text{srt}}\boldsymbol{\Phi}_{\text{sparse}}$ . Therefore, the bound on  $\|\tilde{\mathbf{x}} - \mathbf{x}\|_2$  follows immediately from Lemma 1 and Theorem 3 which shows that  $\mathbf{S} = \boldsymbol{\Phi}_{\text{srt}}\boldsymbol{\Phi}_{\text{sparse}}$  is an  $(r, \delta, \epsilon/4)$ -OSE. And the running time analysis of Algorithm 1 is given in Section 3. □

## 4 RISK INFLATION BOUND

In this section, we study the risk inflation of the approximate solution  $\tilde{\mathbf{x}}$  returned by Algorithm 1 with respect to the optimal solution  $\mathbf{x}^*$  of ridge regression. We begin with review the definition of risk of ridge regression. To properly define the risk, we need to that  $\mathbf{A}$  and  $\mathbf{b}$  have the linear relationship as follows

$$\mathbf{b} = \mathbf{A}\mathbf{x}_0 + \mathbf{e}, \tag{12}$$

where  $\mathbf{x}_0 \in \mathbb{R}^p$  is an unknown vector which is assumed to be the “true” parameter and  $\mathbf{e} \in \mathbb{R}^n$  is independent noise in each coordinate, with  $\mathbf{E}[e_i] = 0$  and  $\mathbf{Var}[e_i] = \sigma^2$ . Under this assumption, the risk of any vector  $\hat{\mathbf{b}} \in \mathbb{R}^n$  is given by

$$\text{risk}(\hat{\mathbf{b}}) \triangleq \frac{1}{n} \mathbf{E} \left[ \|\hat{\mathbf{b}} - \mathbf{A}\mathbf{x}_0\|_2^2 \right],$$

where the expectation is taken over the randomness of noise [4].

The following theorem shows that, compared with the optimal solution  $\mathbf{x}^*$ , the approximate solution  $\tilde{\mathbf{x}}$  returned by Algorithm 1 increases the risk by a small additive factor.

**Theorem 6.** *Given  $\mathbf{A} \in \mathbb{R}^{n \times p}$  of rank  $r$ ,  $\mathbf{b} \in \mathbb{R}^n$ ,  $\lambda > 0$ ,  $\epsilon \in (0, 1)$  and  $\lambda \in (0, 1)$ . Assume that  $\mathbf{A}$  and  $\mathbf{b}$  have the linear relationship as in Eq. (12). Let  $\tilde{\mathbf{x}}$  denote the output of Algorithm 1 with inputs  $\mathbf{A}$ ,  $\mathbf{b}$ ,  $\lambda$ ,  $t' = \lceil 2\delta^{-1}(r^2 + r)/(\epsilon/6 - \epsilon^2/144)^2 \rceil$  and  $t = \lceil 72\epsilon^{-1}[\sqrt{r} + \sqrt{8\log(6p/\delta)}]^2 \log(6r/\delta) \rceil$ . Let  $\mathbf{x}^*$  denote the optimal solution of ridge regression. Then, with probability at least  $1 - \delta$ ,*

$$\text{risk}(\tilde{\mathbf{b}}) \leq \text{risk}(\mathbf{b}^*) + \frac{3\epsilon}{n} \|\mathbf{A}\|_2^2 \left( \|\mathbf{x}_0\|_2^2 + \sigma^2 \rho^2 \right), \tag{13}$$

where we define  $\tilde{\mathbf{b}} = \mathbf{A}\tilde{\mathbf{x}}$  and  $\mathbf{b}^* = \mathbf{A}\mathbf{x}^*$ ; we also define  $\rho^2 = \sum_{i \in [r]} \left( \frac{\sigma_i}{\sigma_i^2 + \lambda} \right)^2$  and  $\sigma_i$  is the  $i$ -th largest singular value of  $\mathbf{A}$ .

## 5 EXTENSIONS

In this section, we present two extensions to our algorithm. First, we consider the multiple response ridge regression



problem, and obtain an efficient approximation algorithm with relative-error guarantee similar with Algorithm 1. Second, combining with the recent results of Avron et al. [3], we present a fast relative-error approximation algorithm of a special nonlinear ridge regression problem called structured ridge regression.

## 5.1 MULTIPLE RESPONSE RIDGE REGRESSION

In this part, we generalize our techniques to solve multiple response ridge regression [11]. The multiple response ridge regression problem is defined as follows. Given a design matrix  $\mathbf{A} \in \mathbb{R}^{n \times p}$ ,  $m$  target vectors (responses)  $\mathbf{B} \in \mathbb{R}^{p \times m}$  and a regression parameter  $\lambda > 0$ , the multiple response regression problem is to find an  $n \times m$  matrix  $\mathbf{X}^*$  such that

$$\mathbf{X}^* = \arg \min_{\mathbf{X} \in \mathbb{R}^{n \times m}} \|\mathbf{A}\mathbf{X} - \mathbf{B}\|_F^2 + \lambda \|\mathbf{X}\|_F^2. \quad (14)$$

The optimal solution of Eq. (14) is given by

$$\mathbf{X}^* = \mathbf{A}^T (\lambda \mathbf{I}_n + \mathbf{A}\mathbf{A}^T)^{-1} \mathbf{B}. \quad (15)$$

It is clear that Eq. (15) takes  $O(n^2p + n^3 + nm)$  time to compute, which is expensive if  $p \gg n \gg 1$ .

Next, we generalize our techniques to solve multiple response regression problem. The first step is to compute the sketched matrix  $\mathbf{A}\mathbf{S}^T$ , where  $\mathbf{S} = \Phi_{\text{srht}} \Phi_{\text{sparse}}$ . Notice that this step is identical to that of Algorithm 1, which uses one pass through  $\mathbf{A}$ . Then, we use the following generalized version of Eq. (3) to compute the approximate solution  $\tilde{\mathbf{X}}$ ,

$$\tilde{\mathbf{X}} = \mathbf{A}^T (\mathbf{A}\mathbf{S}^T)^\dagger (\lambda (\mathbf{A}\mathbf{S}^T)^\dagger + \mathbf{A}\mathbf{S}^T)^\dagger \mathbf{B}, \quad (16)$$

which uses a second pass through  $\mathbf{A}$ . We show that the approximate solution  $\tilde{\mathbf{X}}$  given by Eq. (16) is a relative-error approximation of  $\mathbf{X}^*$  in the following theorem.

**Theorem 7.** *Given  $\mathbf{A} \in \mathbb{R}^{n \times p}$  of rank  $r$ ,  $\mathbf{B} \in \mathbb{R}^{n \times m}$ ,  $\lambda > 0$ , parameter  $\epsilon \in (0, 1)$  and  $\delta \in (0, 1)$ . Select integers  $t', t$  such that  $t' \geq 2\delta^{-1}(r^2 + r)/(\epsilon/6 - \epsilon^2/144)^2$  and  $t \geq 72\epsilon^{-1}[\sqrt{r} + \sqrt{8 \log(6p/\delta)}]^2 \log(6r/\delta)$ . Let  $\mathbf{S} = \Phi_{\text{srht}} \Phi_{\text{sparse}}$ , where  $\Phi_{\text{sparse}} \in \mathbb{R}^{t' \times p}$  is a sparse embedding matrix and  $\Phi_{\text{srht}} \in \mathbb{R}^{t \times t'}$  is an SRHT matrix. Then, with probability at least  $1 - \delta$ , we have*

$$\left\| \tilde{\mathbf{X}} - \mathbf{X}^* \right\|_F \leq \epsilon \|\mathbf{X}^*\|_F,$$

where  $\tilde{\mathbf{X}}$  is given by Eq. (16) and  $\mathbf{X}^*$  is the optimal solution to multiple response ridge regression Eq. (14). In addition, the total time complexity of computing  $\mathbf{A}\mathbf{S}^T$  and  $\tilde{\mathbf{X}}$  is  $O(\text{nnz}(\mathbf{A}) + nr^2 \log(\frac{r}{\epsilon})/\epsilon^2 + n^2r \log(r)/\epsilon + nm)$ .

## 5.2 STRUCTURED RIDGE REGRESSION

In this part, we consider a non-linear ridge regression problem, called *structured ridge regression*, which is closely related to kernel ridge regression with polynomial kernels.

Structured ridge regression uses a non-linear kernel expansion function  $\varphi_q$ , which is studied recently by Avron et al. [3] under the context of (non-regularized) structured regression. The kernel expansion function  $\varphi_q$  maps a  $p$ -dimensional vector  $\mathbf{a}$  to a  $pq$ -dimensional vector  $\varphi_q(\mathbf{a}) = \{a_i^{j-1}\}_{(i,j) \in [p] \times [q]}$  for  $q > 1$ . The definition of  $\varphi_q$  corresponds to the kernel function  $k_{\varphi_q}(\mathbf{a}, \mathbf{b}) = \varphi_q(\mathbf{a})^T \varphi_q(\mathbf{b}) = \sum_{(i,j) \in [p] \times [q]} (a_i b_j)^{j-1}$ , for any  $p$ -dimensional  $\mathbf{a}$  and  $\mathbf{b}$ . Clearly,  $k_{\varphi_q}$  is related to polynomial kernels, which is defined as  $k_q(\mathbf{a}, \mathbf{b}) = (\mathbf{a}^T \mathbf{b})^q = \left( \sum_{i \in [p]} a_i b_i \right)^q$ . For more detailed discussion and the connections of  $\varphi_q$  to other kernels, we refer interested readers to [3].

In the following, we define structured ridge regression, which can be seen an  $l_2$  regularized version of structured regression proposed by Avron et al. [3],

$$\mathbf{x}^* = \arg \min_{\mathbf{x} \in \mathbb{R}^{pq}} \|\varphi_q(\mathbf{A})\mathbf{x} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{x}\|_2^2, \quad (17)$$

where  $\varphi_q(\mathbf{A})$  is an  $n \times pq$  matrix consisting of  $n$  expanded samples, i.e. its  $i$ -th row vector is  $\varphi_q(\mathbf{A})_{(i)} = \varphi_q(\mathbf{A}_{(i)})$  for all  $i \in [n]$ . Clearly, Eq (17) is a non-linear ridge regression problem. For this problem, the dual space approach gives that  $\mathbf{x}^* = \varphi_q(\mathbf{A})^T (\mathbf{K} + \lambda \mathbf{I}_n)^{-1} \mathbf{b}$ , where  $\mathbf{K} = \varphi_q(\mathbf{A}) \varphi_q(\mathbf{A})^T$ . It is clear that computing  $\mathbf{x}^*$  using this approach takes  $O(n^2pq + n^3)$  time.

We extend our techniques to accelerate the computation of structured ridge regression for  $p \gg n$  instances by using the following property of  $\varphi_q$ . Avron et al. [3] showed that there exists a fast multiplication algorithm which computes the product  $\varphi_q(\mathbf{A}) \Phi_{\text{sparse}}^T$  in  $O((\text{nnz}(\mathbf{A}) + nqt') \log^2(q))$  time by exploiting the structure of  $\varphi_q(\mathbf{A})$ . Therefore, we only need to modify Algorithm 1 to use the fast multiplication algorithm for computing the sketched matrix  $(\varphi_q(\mathbf{A}) \Phi_{\text{sparse}}^T) \Phi_{\text{srht}}^T$ ; then run the modified algorithm with input  $\varphi_q(\mathbf{A})$  and  $\mathbf{b}$ . We show that this procedure gives a relative-error approximation algorithm for structured ridge regression that runs in  $O(\text{nnz}(\mathbf{A}) \log^2(q) + n^3q \log^2(q)/\epsilon^2 + n^3 \log(\frac{n}{\epsilon})/\epsilon^2)$  time, which is faster than the dual space approach when  $p \gg n$ . For constant  $q$ , this is also asymptotically faster than solving kernel ridge regression with polynomial kernel in dual space, whose computational complexity is  $O(n^2p + n^3)$ . We defer the detailed description of the approximation algorithm and its related analysis to the supplementary material (see Section D).

## 6 EXPERIMENTS

**Baselines.** We compare the performance of our algorithm SKETCHING (Algorithm 1) to three baselines. The first baseline is the STANDARD algorithm, which computes the optimal solution using the dual space approach in Eq. (2). The other two baselines use popular randomized dimen-

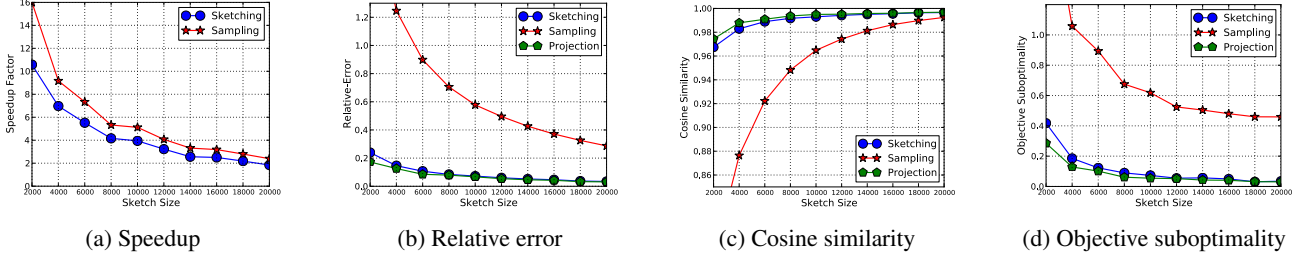


Figure 1: Quality-of-approximation and running times on synthetic dataset

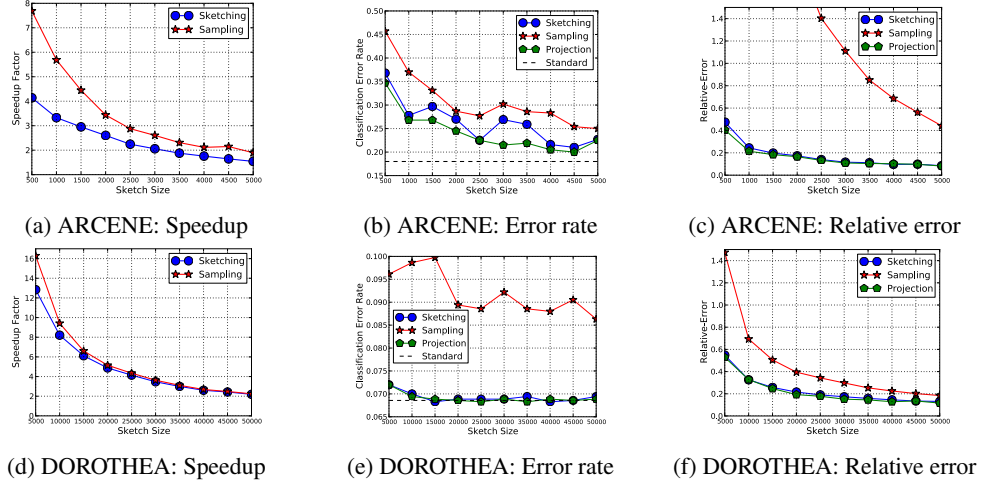


Figure 2: Classification accuracy and running times on realworld datasets

sionality reduction methods, including sampling and random projection. The SAMPLING algorithm simply samples a subset of  $t$  columns of  $\mathbf{A}$  uniformly at random. The PROJECTION algorithm post-multiplies  $\mathbf{A}$  by a random sign matrix  $\Phi_{\text{sign}}^T \in \mathbb{R}^{p \times t}$ , with each entry of  $\Phi_{\text{sign}}$  having value chosen from  $\{\pm 1/\sqrt{t}\}$  uniformly at random. Then, sketched matrices with  $t$  columns obtained by SAMPLING and PROJECTION are plugged in Eq. (3) to compute an approximate solution  $\tilde{\mathbf{x}}$  of ridge regression. Finally, notice that  $\Phi_{\text{sign}}$  is also an OSE for sufficiently large  $t$  [28] and therefore, by Lemma 1, the PROJECTION algorithm produces a relative-error approximation as well. However, for dense  $\mathbf{A}$ , computing  $\mathbf{A}\Phi_{\text{sign}}^T$  alone takes  $O(tnp)$  time, which is even slower than the STANDARD algorithm when  $t > n$ . Hence, we do not compare its running time to other competing algorithms.

**Implementation.** Our implementation of Algorithm 1 is slightly different from its description in two places. First, Algorithm 1, and its analysis, uses the Walsh-Hadamard transformation (as a step of SRHT), while our implementation uses discrete Hartley transformation (DHT) [30]. DHT has a highly optimized implementation provided in FFTW package [17]. In addition, it is possible to show that DHT or other Fourier-type transformations have a guarantee similar to Walsh-Hadamard transformation [2, 32]. Second, we set

$t' = 2t$ , i.e. the sketch size of sparse embedding is two times larger than that of SRHT. Empirically, this setting offers a good trade-off between accuracy and speed. All competing algorithms are implemented using C++ and the experiments are conducted on a standard workstation using a single core.

## 6.1 SYNTHETIC DATASET

**Setup.** We generate the  $n \times p$  design matrix  $\mathbf{A}$  using the following method, such that each row (sample) of  $\mathbf{A}$  contains an  $s$ -dimensional signal and  $p$ -dimensional noises. Specifically, we define  $\mathbf{A} = \mathbf{M}\mathbf{\Sigma}\mathbf{V}^T + \alpha\mathbf{E}$ . Here,  $\mathbf{M}$  is an  $n \times s$  matrix which represents the signals, and each entry  $M_{ij} \sim \mathcal{N}(0, 1)$  is an i.i.d Gaussian random variable.  $\mathbf{\Sigma}$  is an  $s \times s$  diagonal matrix and the diagonal entries are given by  $\Sigma_{ii} = 1 - (i - 1)/p$  for each  $i \in [s]$ .  $\mathbf{V}$  is a  $p \times n$  column orthonormal matrix which contains a random  $s$ -dimensional subspace of  $\mathbb{R}^p$ . Notice that  $\mathbf{M}\mathbf{\Sigma}\mathbf{V}^T$  is a rank  $s$  matrix with linearly decreasing singular values.  $\mathbf{E}$  is an  $n \times p$  matrix which contributes the additive Gaussian noise  $E_{ij} \sim \mathcal{N}(0, 1)$ .  $\alpha > 0$  is a parameter chosen to balance the energy of signals  $\mathbf{M}\mathbf{\Sigma}\mathbf{V}^T$  and the energy of noises  $\mathbf{E}$ . In this experiment, we choose  $\alpha = 0.05$  which brings  $\|\mathbf{M}\mathbf{\Sigma}\mathbf{V}^T\|_F \approx \alpha \|\mathbf{E}\|_F$ . Then, we generate the target vector  $\mathbf{x} \in \mathbb{R}^p$  with  $x_i \sim \mathcal{N}(0, 1)$ . Finally, the target vector

$\mathbf{b} \in \mathbb{R}^n$  is given by  $\mathbf{b} = \mathbf{A}\mathbf{x} + \gamma\mathbf{e}$ , where  $e_i \sim \mathcal{N}(0, 1)$  and  $\gamma = 5$ .

**Metrics.** We measure the performance of our algorithm and baselines both in terms of accuracy and speedup factor. More specifically, let  $\mathbf{x}^*$  denote the optimal solution produced by the standard algorithm and let  $\tilde{\mathbf{x}}$  denote the output vector returned by an approximation algorithm. To evaluate the accuracy of approximation, we compute three metrics: *relative error*:  $\frac{\|\tilde{\mathbf{x}} - \mathbf{x}^*\|_2}{\|\mathbf{x}^*\|_2}$ ; *cosine similarity*:  $\frac{\tilde{\mathbf{x}}^T \mathbf{x}^*}{\|\tilde{\mathbf{x}}\|_2 \|\mathbf{x}^*\|_2}$ ; *objective suboptimality*:  $\frac{f(\tilde{\mathbf{x}})}{f(\mathbf{x}^*)} - 1$ , with  $f(\mathbf{x}) \triangleq \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{x}\|_2^2$ . In addition, the speedup factor is given by the ratio between the time used by STANDARD algorithm and that of a competing algorithm.

**Results.** In the experiment, we set  $n = 500$ ,  $p = 50000$  and  $s = 50$ . We run the competing algorithms with 10 different choices of  $t$  within range  $[2000, 20000]$ . The results are shown in Figure 1. Figure 1(a) reports the speed up of approximation algorithms with respect to the STANDARD algorithm. We see that our algorithm SKETCHING is slightly slower than the SAMPLING algorithm, but both of them speed up considerably with respect to the STANDARD algorithm. Figure 1(b), (c) and (d) plot the accuracy metrics of the competing algorithms. We see that indeed the accuracy of approximation improves as the sketch size  $t$  increases. In addition, both of our SKETCHING algorithm and the PROJECTION algorithm output a significantly more accurate solution than the SAMPLING algorithm. Notably, when the sketch size  $t \approx 10000$ , our algorithm SKETCHING has a relative-error smaller than 10%, cosine similarity larger than 99% and objective suboptimality less than 10%; meanwhile speeds up the computation about 4 times.

## 6.2 REALWORLD DATASETS

**Setup.** We also test the proposed algorithm on two binary classification datasets: ARCENE and DOROTHEA [18]. Both datasets are publicly available from the UCI repository [6]. ARCENE contains 200 samples (100 for training and 100 for testing) with 10000 real valued features. DOROTHEA consists of 1150 samples (800 for training and 350 for testing) with 100000 binary valued features. We apply ridge regression on both classification tasks by setting the responses to be +1 for positive examples and -1 for negative examples. We run ridge regression algorithms on the training data to compute the feature weights and measure the classification error rate on the testing data. For each dataset, we test 10 different choices of sketch size  $t$  and record the classification error rates and speed up factors of competing algorithms.

**Results.** The experiment results are shown in Figure 2. From the results, we observe that the classification error decreases as the sketch size  $t$  increases. It is also clear that, using the same sketch size  $t$ , SKETCHING and PRO-

JECTION produce more accurate predictions than SAMPLING. On the other hand, SKETCHING and SAMPLING algorithms are considerably faster than the STANDARD algorithm. From the results, we see that our algorithm SKETCHING substantially speeds up the computation, while attains a very small increase in error rate. For ARCENE dataset, when  $t \approx 3000$ , SKETCHING accelerates the computation by 2.1 times while increases the error rate by 4.5%; and, for DOROTHEA dataset, when  $t \approx 20000$ , the speedup of SKETCHING is about 4.1 times and the error rate is almost the same to the STANDARD algorithm. In addition, we continue to observe that the relative-error decreases as  $t$  increases. The SKETCHING algorithm and the PROJECTION algorithm outperform the SAMPLING algorithm in terms of accuracy on both datasets. We remark that, for moderately large  $t$ , the SKETCHING algorithm achieves a relative-error that is smaller than 20% on both datasets.

## 7 CONCLUSIONS

We presented an efficient relative-error approximation algorithm for ridge regression for  $p \gg n$  cases. Our algorithm runs in  $\tilde{O}(\text{nnz}(A) + n^3/\epsilon^2)$  time, which is substantially faster than the existing  $O(n^2p + n^3)$  algorithm for large  $p$  instances. In addition, we analyzed the risk inflation of our algorithm and extended our techniques to design fast relative-error approximation algorithms for multiple response ridge regression and structured ridge regression. We reported experimental results of our algorithm on both synthetic and real datasets, which supported our analysis and demonstrated good practical performance.

### Acknowledgments

The work described in this paper was fully supported by the National Grand Fundamental Research 973 Program of China (No. 2014CB340401 and No. 2014CB340405), the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CUHK 413213 and CUHK 14205214), Microsoft Research Asia Regional Seed Fund in Big Data Research (Grant No. FY13-RESPONSOR-036) and Research Grants Council of the Hong Kong S.A.R. (Project no. CUHK419413).

### References

- [1] Ahmed El Alaou and Michael W. Mahoney. Fast randomized kernel methods with statistical guarantees. *Technical Report*, 2014.
- [2] Haim Avron, Christos Boutsidis, Sivan Toledo, and Anastasios Zouzias. Efficient dimensionality reduction for canonical correlation analysis. In *ICML*, 2013.

- [3] Haim Avron, Vikas Sindhvani, and David Woodruff. Sketching structured matrices for faster nonlinear regression. In *NIPS*, 2013.
- [4] Francis Bach. Sharp analysis of low-rank kernel matrix approximations. In *COLT*, 2013.
- [5] Francis R Bach and Michael I Jordan. Kernel independent component analysis. *JMLR*, 2003.
- [6] K. Bache and M. Lichman. UCI machine learning repository, 2013.
- [7] Mohamed-Ali Belabbas and Patrick J Wolfe. Spectral methods in machine learning and new strategies for very large datasets. *PNAS*, 2009.
- [8] Jean Bourgain and Jelani Nelson. Toward a unified theory of sparse dimensionality reduction in euclidean space. *Technical Report*, 2013.
- [9] C. Boutsidis and M. Magdon-Ismail. Faster svd-truncated regularized least-squares. In *ISIT*, 2014.
- [10] Christos Boutsidis, Anastasios Zouzias, and Petros Drineas. Random projections for  $k$ -means clustering. In *NIPS*. 2010.
- [11] Leo Breiman and Jerome H Friedman. Predicting multivariate responses in multiple linear regression. *J R STAT SOC B*, 1997.
- [12] Kenneth L Clarkson and David P Woodruff. Low rank approximation and regression in input sparsity time. In *STOC*, 2013.
- [13] Corinna Cortes, Mehryar Mohri, and Ameet Talwalkar. On the impact of kernel approximation on learning accuracy. In *AISTATS*, 2010.
- [14] Sanjoy Dasgupta and Anupam Gupta. An elementary proof of a theorem of johnson and lindenstrauss. *RS&A*, 2003.
- [15] Petros Drineas and Michael W Mahoney. On the nystrom method for approximating a gram matrix for improved kernel-based learning. *JMLR*, 2005.
- [16] Shai Fine and Katya Scheinberg. Efficient svm training using low-rank kernel representations. *JMLR*, 2002.
- [17] Matteo Frigo and Steven G. Johnson. The design and implementation of FFTW3. *P IEEE*, 2005.
- [18] Isabelle Guyon. Design of experiments of the nips 2003 variable selection benchmark. In *NIPS 2003 workshop on feature extraction and feature selection*, 2003.
- [19] Sanjiv Kumar, Mehryar Mohri, and Ameet Talwalkar. On sampling-based approximate spectral decomposition. In *ICML*, 2009.
- [20] Sanjiv Kumar, Mehryar Mohri, and Ameet Talwalkar. Sampling techniques for the nystrom method. In *AISTATS*, 2009.
- [21] Quoc Le, Tamas Sarlos, and Alexander Smola. Fastfood-computing hilbert space expansions in logarithmic time. In *ICML*, 2013.
- [22] Yichao Lu, Paramveer Dhillon, Dean P Foster, and Lyle Ungar. Faster ridge regression via the subsampled randomized hadamard transform. In *NIPS*, 2013.
- [23] Michael W Mahoney, Petros Drineas, Malik Magdon-Ismail, and David P Woodruff. Fast approximation of matrix coherence and statistical leverage. In *ICML*, 2012.
- [24] Jelani Nelson and Huy L. Nguyen. Osnap: Faster numerical linear algebra algorithms via sparser subspace embeddings. In *FOCS*, 2013.
- [25] Saurabh Paul, Christos Boutsidis, Malik Magdon-Ismail, and Petros Drineas. Random projections for support vector machines. In *AISTATS*, 2013.
- [26] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *NIPS*, 2007.
- [27] Garvesh Raskutti and Michael Mahoney. Statistical and algorithmic perspectives on randomized sketching for ordinary least-squares. In *ICML*, 2015.
- [28] Tamas Sarlós. Improved approximation algorithms for large matrices via random projections. In *FOCS*, 2006.
- [29] Craig Saunders, Alexander Gammerman, and Volodya Vovk. Ridge regression learning algorithm in dual variables. In *ICML*, 1998.
- [30] H.V. Sorensen, D.L. Jones, C.S. Burrus, and M. Heidemman. On computing the discrete hartley transform. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 1985.
- [31] Terence Tao. *Topics in random matrix theory*. 2012.
- [32] Joel A Tropp. Improved analysis of the subsampled randomized hadamard transform. *AADA*, 2011.
- [33] Christopher Williams and Matthias Seeger. Using the nystrom method to speed up kernel machines. In *NIPS*, 2001.
- [34] Kai Zhang, Ivor W Tsang, and James T Kwok. Improved nystrom low-rank approximation and error analysis. In *ICML*, 2008.
- [35] Yuchen Zhang, John Duchi, and Martin Wainwright. Divide and conquer kernel ridge regression. In *COLT*, 2013.

---

# Selective Greedy Equivalence Search: Finding Optimal Bayesian Networks Using a Polynomial Number of Score Evaluations

---

David Maxwell Chickering  
Microsoft Research  
Redmond, WA 98052  
dmax@microsoft.com

Christopher Meek  
Microsoft Research  
Redmond, WA 98052  
meek@microsoft.com

## Abstract

We introduce Selective Greedy Equivalence Search (SGES), a restricted version of Greedy Equivalence Search (GES). SGES retains the asymptotic correctness of GES but, unlike GES, has polynomial performance guarantees. In particular, we show that when data are sampled independently from a distribution that is perfect with respect to a DAG  $\mathcal{G}$  defined over the observable variables then, in the limit of large data, SGES will identify the equivalence class of  $\mathcal{G}$  after a number of score evaluations that is (1) polynomial in the number of nodes and (2) exponential in various complexity measures including maximum-number-of-parents, maximum-clique-size, and a new measure called *v-width* that is at least as small as—and potentially much smaller than—the other two. More generally, we show that for any hereditary and equivalence-invariant property  $\Pi$  known to hold in  $\mathcal{G}$ , we retain the large-sample optimality guarantees of GES even if we ignore any GES deletion operator during the backward phase that results in a state for which  $\Pi$  does not hold in the common-descendants subgraph.

## 1 INTRODUCTION

Greedy Equivalence Search (GES) is a score-based search algorithm that searches over equivalence classes of Bayesian-network structures. The algorithm is appealing because (1) for finite data, it explicitly (and greedily) tries to maximize the score of interest, and (2) as the data grows large, it is guaranteed—under suitable distributional assumptions—to return the generative structure. Although empirical results show that the algorithm is efficient in real-world domains, the number of search states that GES needs to evaluate in the worst case can be exponential in the number of domain variables.

In this paper, we show that if we assume the generative distribution is perfect with respect to some DAG  $\mathcal{G}$  defined over the observable variables, and if  $\mathcal{G}$  is known to be constrained by one of various graph-theoretic measures of complexity, then we can disregard all but a polynomial number of the backward search operators considered by GES while retaining the large-sample guarantees of the algorithm; we call this new variant of GES *selective greedy equivalence search* or *SGES*. Our complexity results are a consequence of a new understanding of the backward phase of GES, in which edges (either directed or undirected) are greedily deleted from the current state until a local minimum is reached. We show that for any *hereditary* and *equivalence-invariant* property known to hold in generative model  $\mathcal{G}$ , we can remove from consideration any edge-deletion operator between  $X$  and  $Y$  for which the property does not hold in the resulting induced subgraph over  $X$ ,  $Y$ , and their common descendants. As an example, if we know that each node has at most  $k$  parents, we can remove from consideration any deletion operator that results in a common child with more than  $k$  parents.

By casting limited *v-width* and two other complexity measures—maximum-clique size and maximum-parent-set size—as graph properties, we show how to enumerate directly over a polynomial number of edge-deletion operators at each step, and we show that we need only a polynomial number of calls to the scoring function to complete the algorithm.

The main contributions of this paper are theoretical. Our definition of the new SGES algorithm deliberately leaves unspecified the details of how to implement its forward phase; we prove our results for SGES given *any* implementation of this phase that completes with a polynomial number of calls to the scoring function. A naive implementation is to immediately return a complete (i.e., no independence) graph using *no* calls to the scoring function, but this choice is unlikely to be reasonable in practice, particularly in discrete domains where the sample complexity of this initial model will likely be a problem. Whereas we believe it an important direction, our paper does not explore practical al-

ternatives for the forward phase that have polynomial-time guarantees.

Our paper is organized as follows. In Section 2, we describe related work. In Section 3, we provide notation and background material. In Section 4, we present our new SGES algorithm, we show that it is optimal in the large-sample limit, and we provide complexity bounds when given an equivalence-invariant and hereditary property that holds on the generative structure. In Section 5, we present a simple synthetic experiment that demonstrates the value of restricting the backward operators in SGES. We conclude with a discussion of our results in Section 6.

## 2 RELATED WORK

It is useful to distinguish between approaches to learning the structure of graphical models as *constraint based*, *score based* or *hybrid*. Constraint-based approaches typically use (conditional) independence tests to eliminate potential models, whereas score-based approaches typically use a penalized likelihood or a marginal likelihood to evaluate alternative model structures; hybrid methods combine these two approaches. Because score-based approaches are driven by a global likelihood, they are less susceptible than constraint-based approaches to incorrect categorical decisions about independences.

There are polynomial-time algorithms for learning the best model in which each node has at most one parent. In particular, the Chow-Liu algorithm (Chow and Liu, 1968) used with any equivalence-invariant score will identify the highest-scoring tree-like model in polynomial time; for scores that are not equivalence invariant, we can use the polynomial-time maximum-branching algorithm of Edmonds (1967) instead. Gaspers et al. (2012) show how to learn *k-branchings* in polynomial time; these models are polytrees that differ from a branching by a constant  $k$  number of edge deletions.

Without additional assumptions, most results for learning non-tree-like models are negative. Meek (2001) shows that finding the maximum-likelihood path is NP-hard, despite this being a special case of a tree-like model. Dasgupta (1999) shows that finding the maximum-likelihood polytree (a graph in which each pair of nodes is connected by at most one path) is NP-hard, even with bounded in-degree for every node. For general directed acyclic graphs, Chickering (1996) shows that finding the highest marginal-likelihood structure is NP-hard, even when each node has at most two parents. Chickering et al. (2004) extend this same result to the large-sample case.

Researchers often assume that the “generative” distribution of the training data is *perfect* with respect to some model class in order to reduce the complexity of learning algorithms. Geiger et al. (1990) provide a polynomial-time

constraint-based algorithm for recovering a polytree under the assumption that the generative distribution is perfect with respect to a polytree; an analogous score-based result follows from this paper. The constraint-based PC algorithm of Sprites et al. (1993) can identify the equivalence class of Bayesian networks in polynomial time if the generative structure is a DAG model over the observable variables in which each node has a bounded degree; this paper provides a similar result for a score-based algorithm. Kalish and Buhlmann (2007) show that for Gaussian distributions, the PC algorithm can identify the right structure even when the number of nodes in the domain is larger than the sample size. Chickering (2002) uses the same DAG-perfectness-over-observables assumption to show that the greedy GES algorithm is optimal in the large-sample limit, although the branching factor of GES is worst-case exponential; the main result of this paper shows how to limit this branching factor without losing the large-sample guarantee. Chickering and Meek (2002) show that GES identifies a “minimal” model in the large-sample limit under a less restrictive set of assumptions.

Hybrid methods for learning DAG models use a constraint-based algorithm to prune out a large portion of the search space, and then use a score-based algorithm to select among the remaining (Friedman et al., 1999; Tsamardinos et al., 2006). Ordyniak and Szeider (2013) give positive complexity results for the case when the remaining DAGs are characterized by a structure with constant treewidth.

Many researchers have turned to exhaustive enumeration to identify the highest-scoring model (Gillispie and Perlman, 2001; Koivisto and Sood 2004; Silander and Myllymäki, 2006; Kojima et al, 2010). There are many complexity results for other model classes. Karger and Srebro (2001) show that finding the optimal Markov network is NP-complete for treewidth  $> 1$ . Narasimhan and Bilmes (2004) and Shahaf, Chechetka and Guestrin (2009) show how to learn approximate limited-treewidth models in polynomial time. Abeel, Koller and Ng (2005) show how to learn factor graphs in polynomial time.

## 3 NOTATION AND BACKGROUND

We denote a variable by an upper case letter (e.g.,  $A$ ) and a state or value of that variable by the same letter in lower case (e.g.,  $a$ ). We denote a set of variables by a bold-face capitalized letter or letters (e.g.,  $\mathbf{X}$ ). We use a corresponding bold-face lower-case letter or letters (e.g.,  $\mathbf{x}$ ) to denote an assignment of state or value to each variable in a given set. We use calligraphic letters (e.g.,  $\mathcal{G}, \mathcal{E}$ ) to denote statistical models and graphs.

A *Bayesian-network model* for a set of variables  $U$  is a pair  $(\mathcal{G}, \theta)$ .  $\mathcal{G} = (\mathbf{V}, \mathbf{E})$  is a directed acyclic graph—or *DAG* for short—consisting of nodes in one-to-one correspondence with the variables and directed edges that connect

those nodes.  $\theta$  is a set of parameter values that specify all of the conditional probability distributions. The Bayesian network represents a joint distribution over  $\mathbf{U}$  that factors according to the structure  $\mathcal{G}$ .

The structure  $\mathcal{G}$  of a Bayesian-network model represents the independence constraints that must hold in the distribution. The set of all independence constraints implied by the structure  $\mathcal{G}$  can be characterized by the *Markov conditions*, which are the constraints that each variable is independent of its non-descendants given its parents. All other independence constraints follow from properties of independence. A distribution defined over the variables from  $\mathcal{G}$  is *perfect with respect to  $\mathcal{G}$*  if the set of independences in the distribution is equal to the set of independences implied by the structure  $\mathcal{G}$ .

Two DAGs  $\mathcal{G}$  and  $\mathcal{G}'$  are *equivalent*<sup>1</sup>—denoted  $\mathcal{G} \approx \mathcal{G}'$ —if the independence constraints in the two DAGs are identical. Because equivalence is reflexive, symmetric, and transitive, the relation defines a set of equivalence classes over network structures. We use  $[\mathcal{G}]_{\approx}$  to denote the equivalence class of DAGs to which  $\mathcal{G}$  belongs.

An equivalence class of DAGs  $\mathcal{F}$  is an *independence map (IMAP)* of another equivalence class of DAGs  $\mathcal{E}$  if all independence constraints implied by  $\mathcal{F}$  are also implied by  $\mathcal{E}$ . For two DAGs  $\mathcal{G}$  and  $\mathcal{H}$ , we use  $\mathcal{G} \leq \mathcal{H}$  to denote that  $[\mathcal{H}]_{\approx}$  is an IMAP of  $[\mathcal{G}]_{\approx}$ ; we use  $\mathcal{G} < \mathcal{H}$  when  $\mathcal{G} \leq \mathcal{H}$  and  $[\mathcal{H}]_{\approx} \neq [\mathcal{G}]_{\approx}$ .

Verma and Pearl (1991) show that two DAGs are equivalent if and only if they have the same *skeleton* (i.e., the graph resulting from ignoring the directionality of the edges) and the same *v-structures* (i.e., pairs of edges  $X \rightarrow Y$  and  $Y \leftarrow Z$  where  $X$  and  $Z$  are not adjacent). As a result, we can use a *partially directed acyclic graph*—or *PDAG* for short—to represent an equivalence class of DAGs: for a PDAG  $\mathcal{P}$ , the equivalence class of DAGs is the set that has the same skeleton and the same v-structures as  $\mathcal{P}$ <sup>2</sup>.

We extend our notation for DAG equivalence and the DAG IMAP relation to include the more general PDAG structure. In particular, for a PDAG  $\mathcal{P}$ , we use  $[\mathcal{P}]_{\approx}$  to denote the corresponding equivalence class of DAGs. For any pair of PDAGs  $\mathcal{P}$  and  $\mathcal{Q}$ —where one or both may be a DAG—we use  $\mathcal{P} \approx \mathcal{Q}$  to denote  $[\mathcal{Q}]_{\approx} = [\mathcal{P}]_{\approx}$  and we use  $\mathcal{P} \leq \mathcal{Q}$  to denote  $[\mathcal{Q}]_{\approx}$  is an IMAP of  $[\mathcal{P}]_{\approx}$ . To avoid confusion, for the remainder of the paper we will reserve the symbols  $\mathcal{G}$  and  $\mathcal{H}$  for DAGs.

For any PDAG  $\mathcal{P}$  and subset of nodes  $\mathbf{V}$ , we use  $\mathcal{P}[\mathbf{V}]$  to denote the subgraph of  $\mathcal{P}$  induced by  $\mathbf{V}$ ; that is,  $\mathcal{P}[\mathbf{V}]$  has

<sup>1</sup>We make the standard conditional-distribution assumptions of multinomials for discrete variables and Gaussians for continuous variables so that if two DAGs have the same independence constraints, then they can also model the same set of distributions.

<sup>2</sup>The definitions for the skeleton and set of v-structures for a PDAG are the obvious extensions to these definitions for DAGs.

as nodes the set  $\mathbf{V}$  and has as edges all those from  $\mathcal{P}$  that connect nodes in  $\mathbf{V}$ . We use  $\text{NA}_{X,Y}$  to denote, within a PDAG, the set of nodes that are *neighbors* of  $X$  (i.e., connected with an undirected edge) and also adjacent to  $Y$  (i.e., without regard to whether the connecting edge is directed or undirected).

An edge in  $\mathcal{G}$  is *compelled* if it exists in every DAG that is equivalent to  $\mathcal{G}$ . If an edge in  $\mathcal{G}$  is not compelled, we say that it is *reversible*. A *completed PDAG (CPDAG)*  $\mathcal{C}$  is a PDAG with two additional properties: (1) for every directed edge in  $\mathcal{C}$ , the corresponding edge in  $\mathcal{G}$  is compelled and (2) for every undirected edge in  $\mathcal{C}$  the corresponding edge in  $\mathcal{G}$  is reversible. Unlike non-completed PDAGs, the CPDAG representation of an equivalence class is unique. We use  $\text{Pa}_Y^{\mathcal{P}}$  to denote the *parents* of node  $Y$  in  $\mathcal{P}$ . An edge  $X \rightarrow Y$  is *covered* in a DAG if  $X$  and  $Y$  have the same parents, with the exception that  $X$  is not a parent of itself.

### 3.1 Greedy Equivalence Search

---

Algorithm *GES*( $\mathbf{D}$ )

---

**Input** : Data  $\mathbf{D}$   
**Output**: CPDAG  $\mathcal{C}$

$\mathcal{C} \leftarrow \text{FES}(\mathbf{D})$   
 $\mathcal{C} \leftarrow \text{BES}(\mathbf{D}, \mathcal{C})$

**return**  $\mathcal{C}$

---

Figure 1: Pseudo-code for the GES algorithm.

The GES algorithm, shown in Figure 1, performs a two-phase greedy search through the space of DAG equivalence classes. GES represents each search state with a CPDAG, and performs transformation operators to this representation to traverse between states. Each operator corresponds to a DAG edge modification, and is scored using a DAG scoring function that we assume has three properties. First, we assume the scoring function is *score equivalent*, which means that it assigns the same score to equivalent DAGs. Second, we assume the scoring function is *locally consistent*, which means that, given enough data, (1) if the current state *is not* an IMAP of  $\mathcal{G}$ , the score prefers edge additions that remove incorrect independences, and (2) if the current state *is* an IMAP of  $\mathcal{G}$ , the score prefers edge deletions that remove incorrect dependences. Finally, we assume the scoring function is *decomposable*, which means we can express it as:

$$\text{Score}(\mathcal{G}, \mathbf{D}) = \sum_{i=1}^n \text{Score}(X_i, \text{Pa}_i^{\mathcal{G}}) \quad (1)$$

Note that the data  $\mathbf{D}$  is implicit in the right-hand side Equation 1. Most scores in the literature have these properties.

For the remainder of this paper, we assume they hold for our scoring function.

All of the CPDAG operators from GES are scored using differences in the DAG scoring function, and in the limit of large data, these scores are positive precisely for those operators that remove incorrect independences and incorrect dependences.

The first phase of the GES—called *forward equivalence search* or *FES*—starts with an empty (i.e., no-edge) CPDAG and greedily applies *GES insert* operators until no operator has a positive score; these operators correspond precisely to the union of all single-edge additions to all DAG members of the current (equivalence-class) state. After FES reaches a local maximum, GES switches to the second phase—called *backward equivalence search* or *BES*—and greedily applies *GES delete* operators until no operator has a positive score; these operators correspond precisely to the union of all single-edge deletions from all DAG members of the current state.

**Theorem 1. (Chickering, 2002)** *Let  $\mathcal{C}$  be the CPDAG that results from applying the GES algorithm to  $m$  records sampled from a distribution that is perfect with respect to DAG  $\mathcal{G}$ . Then in the limit of large  $m$ ,  $\mathcal{C} \approx \mathcal{G}$ .*

The role of FES in the large-sample limit is only to identify a state  $\mathcal{C}$  for which  $\mathcal{G} \leq \mathcal{C}$ ; Theorem 1 holds for GES under any implementation of FES that results in an IMAP of  $\mathcal{G}$ . The implementation details can be important in practice because what constitutes a “large” amount of data depends on the number of parameters in the model. In theory, however, we could simply replace FES with a (constant-time) algorithm that sets  $\mathcal{C}$  to be the no-independence equivalence class.

The focus of our analysis in the next section is on a modified version of BES, and the details of the delete operator used in this phase are important. In Figure 2, we show the preconditions, scoring function, and transformation algorithm for a delete operator. We note that we do not need to make any CPDAG transformations when *scoring* the operators; it is only once we have identified the highest-scoring (non-negative) delete that we need to make the transformation shown in the figure. After applying the edge modifications described in the **foreach** loop, the resulting PDAG  $\mathcal{P}$  is not necessarily completed and hence we may have to convert  $\mathcal{P}$  into the corresponding CPDAG representation. As shown by Chickering (2002), this conversion can be accomplished easily by using the structure of  $\mathcal{P}$  to extract a DAG that we then convert into a CPDAG by undirecting all reversible edges. The complexity of this procedure for a  $\mathcal{P}$  with  $n$  nodes and  $e$  edges is  $O(n \cdot e)$ , and requires no calls to the scoring function.

---

**Operator:**  $Delete(X, Y, \mathbf{H})$  applied to  $\mathcal{C}$

---

• **Preconditions**

$X$  and  $Y$  are adjacent  
 $\mathbf{H} \subseteq \mathbf{NA}_{Y,X}$   
 $\overline{\mathbf{H}} = \mathbf{NA}_{Y,X} \setminus \mathbf{H}$  is a clique

• **Scoring**

$Score(Y, \{\mathbf{Pa}_Y^{\mathcal{C}} \cup \overline{\mathbf{H}}\} \setminus X) - Score(Y, X \cup \mathbf{Pa}_Y^{\mathcal{C}} \cup \overline{\mathbf{H}})$

• **Transformation**

Remove edge between  $X$  and  $Y$   
**foreach**  $H \in \mathbf{H}$  **do**  
    Replace  $Y - H$  with  $Y \rightarrow H$   
    **if**  $X - H$  **then** Replace with  $X \rightarrow H$ ;  
**end**  
Convert to CPDAG

---

Figure 2: Preconditions, scoring, and transformation algorithm for a delete operator applied to a CPDAG.

## 4 SELECTIVE GREEDY EQUIVALENCE SEARCH

In this section, we define a variant of the GES algorithm called *selective GES*—or *SGES* for short—that uses a subset of the BES operators. The subset is chosen based on a given property  $\Pi$  that is known to hold for the generative structure  $\mathcal{G}$ . Just like GES, SGES—shown in Figure 3—has a forward phase and a backward phase.

For the forward phase of SGES, it suffices for our theoretical analysis that we use a method that returns an IMAP of  $\mathcal{G}$  (in the large-sample limit) using only a polynomial number of insert-operator score calls. For this reason, we call this phase *poly-FES*. A simple implementation of poly-FES is to return the no-independence CPDAG (with no score calls), but other implementations are likely more useful in practice.

The backward phase of SGES—which we call *selective backward equivalence search (SBES)*—uses only a subset of the BES delete operators. This subset must necessarily include all  $\Pi$ -consistent delete operators—defined below—in order to maintain the large-sample consistency of GES, but the subset can (and will) include additional operators for the sake of efficient enumeration.

The DAG properties used by SGES must be *equivalence invariant*, meaning that for any pair of equivalent DAGs, either the property holds for both of them or it holds for neither of them. Thus, for any equivalence-invariant DAG property  $\Pi$ , it makes sense to say that  $\Pi$  either holds or does not hold for a PDAG. As shown by Chickering (1995), a DAG property is equivalence invariant if and only if it is invariant to covered-edge reversals; it follows that the prop-



erty that each node has at most  $k$  parents is equivalence invariant, whereas the property that the length of the longest directed path is at least  $k$  is not. Furthermore, the properties for SGES must also be *hereditary*, which means that if  $\Pi$  holds for a PDAG  $\mathcal{P}$  it must also hold for all induced subgraphs of  $\mathcal{P}$ . For example, the max-parent property is hereditary, whereas the property that each node has *at least*  $k$  parents is not. We use *EIH property* to refer to a property that is equivalence invariant and hereditary.

**Definition 1.  $\Pi$ -Consistent GES Delete**

A GES delete operator  $Delete(X, Y, \mathbf{H})$  is  $\Pi$  consistent for CPDAG  $\mathcal{C}$  if, for the set of common descendants  $\mathbf{W}$  of  $X$  and  $Y$  in the resulting CPDAG  $\mathcal{C}'$ , the property holds for the induced subgraph  $\mathcal{C}'[X \cup Y \cup \mathbf{W}]$ .

In other words, after the delete, the property holds for the subgraph defined by  $X$ ,  $Y$ , and their common descendants.

---

Algorithm  $SGES(\mathbf{D}, \Pi)$

---

**Input** : Data  $\mathbf{D}$ , Property  $\Pi$

**Output**: CPDAG  $\mathcal{C}$

$\mathcal{C} \leftarrow$  poly-FES  
 $\mathcal{C} \leftarrow$  SBES( $\mathbf{D}, \mathcal{C}, \Pi$ )  
**return**  $\mathcal{C}$

---

Figure 3: Pseudo-code for the SGES algorithm.

---

Algorithm  $SBES(\mathbf{D}, \mathcal{C}, \Pi)$

---

**Input** : Data  $\mathbf{D}$ , CPDAG  $\mathcal{C}$ , Property  $\Pi$

**Output**: CPDAG

**Repeat**

**Ops**  $\leftarrow$  Generate  $\Pi$ -consistent delete operators for  $\mathcal{C}$   
 $Op \leftarrow$  highest-scoring operator in **Ops**  
**if** score of  $Op$  is negative **then return**  $\mathcal{C}$   
 $\mathcal{C} \leftarrow$  Apply  $Op$  to  $\mathcal{C}$

---

Figure 4: Pseudo-code for the SBES algorithm.

**4.1 LARGE-SAMPLE CORRECTNESS**

The following theorem establishes a graph-theoretic justification for considering only the  $\Pi$ -consistent deletions at each step of SBES.

**Theorem 2.** *If  $\mathcal{G} < \mathcal{C}$  for CPDAG  $\mathcal{C}$  and DAG  $\mathcal{G}$ , then for any EIH property  $\Pi$  that holds on  $\mathcal{G}$ , there exists a  $\Pi$ -consistent  $Delete(X, Y, \mathbf{H})$  that when applied to  $\mathcal{C}$  results in the CPDAG  $\mathcal{C}'$  for which  $\mathcal{G} \leq \mathcal{C}'$ .*

The proof of Theorem 2 can be found in Chickering and Meek (2015), an expanded version of this paper. The result is a consequence of an explicit characterization of, for a

given pair of DAGs  $\mathcal{G}$  and  $\mathcal{H}$  such that  $\mathcal{G} < \mathcal{H}$ , an edge in  $\mathcal{H}$  that we can either reverse or delete in  $\mathcal{H}$  such that for the resulting DAG  $\mathcal{H}'$ , we have  $\mathcal{G} \leq \mathcal{H}'^3$ .

**Theorem 3.** *Let  $\mathcal{C}$  be the CPDAG that results from applying the SGES algorithm to (1)  $m$  records sampled from a distribution that is perfect with respect to DAG  $\mathcal{G}$  and (2) EIH property  $\Pi$  that holds on  $\mathcal{G}$ . Then in the limit of large  $m$ ,  $\mathcal{C} \approx \mathcal{G}$ .*

**Proof:** Because the scoring function is locally consistent, we know poly-FES must return an IMAP of  $\mathcal{G}$ . Because SBES includes all the  $\Pi$ -consistent delete operators, Theorem 2 guarantees that, unless  $\mathcal{C} \approx \mathcal{G}$ , there will be a positive-scoring operator.  $\square$

**4.2 COMPLEXITY MEASURES**

In this section, we discuss a number of distributional assumptions that we can use with Theorem 3 to limit the number of operators that SGES needs to score. As discussed in Section 2, when we assume the generative distribution is perfect with respect to a DAG  $\mathcal{G}$ , then graph-theoretic assumptions about  $\mathcal{G}$  can lead to more efficient training algorithms. Common assumptions used include (1) a maximum parent-set size for any node, (2) a maximum-clique<sup>4</sup> size among any nodes and (3) a maximum treewidth. Treewidth is important because the complexity of exact inference is exponential in this measure.

We can associate a property with each of these assumptions that holds precisely when the DAG  $\mathcal{G}$  satisfies that assumption. Consider the constraint that the maximum number of parents for any node in  $\mathcal{G}$  is some constant  $k$ . Then, using “PS” to denote parent size, we can define the property  $\Pi_{PS}^k$  to be true precisely for those DAGs in which each node has at most  $k$  parents. Similarly we can define  $\Pi_{CL}^k$  and  $\Pi_{TW}^k$  to correspond to maximum-clique size and maximum treewidth, respectively.

For two properties  $\Pi$  and  $\Pi'$ , we write  $\Pi \subseteq \Pi'$  if for every DAG  $\mathcal{G}$  for which  $\Pi$  holds,  $\Pi'$  also holds. In other words,  $\Pi$  is a *more constraining* property than is  $\Pi'$ . Because the lowest node in any clique has all other nodes in the clique as parents, it is easy to see that  $\Pi_{PS}^k \subseteq \Pi_{CL}^{k-1}$ . Because the treewidth for DAG  $\mathcal{G}$  is defined to be the size of the largest clique minus one in a graph whose cliques are at least as large as those in  $\mathcal{G}$ , we also have  $\Pi_{TW}^k \subseteq \Pi_{CL}^{k-1}$ . Which property to use will typically be a trade-off between how reasonable the assumption is (i.e, less constraining properties are more reasonable) and the efficiency of the resulting algorithm (i.e., more constraining properties lead to faster algorithms).

<sup>3</sup>Chickering (2002) characterizes the *reverse* transformation of reversals/additions in  $\mathcal{G}$ , which provides an *implicit* characterization of reversals/deletions in  $\mathcal{H}$ .

<sup>4</sup>We use *clique* in a DAG to mean a set of nodes in which all pairs are adjacent.

We now consider a new complexity measure called  $v$ -width, whose corresponding property is less constraining than the previous three. For a DAG  $\mathcal{G}$ , the  $v$ -width is defined to be the maximum of, over all pairs of non-adjacent nodes  $X$  and  $Y$ , the size of the largest clique among common children of  $X$  and  $Y$ . In other words,  $v$ -width is similar to the maximum-clique-size bound, except that the bound only applies to cliques of nodes that are shared children of some pair of non-adjacent nodes. With this understanding it is easy to see that, for the property  $\Pi_{VW}^k$  corresponding to a bound on the  $v$ -width, we have  $\Pi_{CL}^k \subseteq \Pi_{VW}^k$ .

To illustrate the difference between  $v$ -width and the other complexity measures, consider the two DAGs in Figure 5. The DAG in Figure 5(a) has a clique of size  $K$ , and consequently a maximum-clique size of  $K$  and a maximum parent-set size of  $K - 1$ . Thus, if  $K$  is  $O(n)$  for a large graph of  $n$  nodes, any algorithm that is exponential in these measures will not be efficient. The  $v$ -width, however, is zero for this DAG. The DAG in Figure 5(b), on the other hand, has a  $v$ -width of  $K$ .

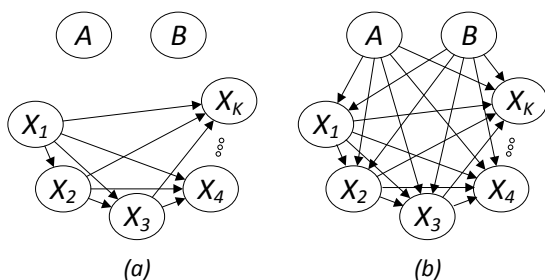


Figure 5: Two DAGs (a) and (b) having identical maximum clique sizes, similar maximum number of parents, and divergent  $v$ -widths.

In order to use a property with SGES, we need to establish that it is EIH. For  $\Pi_{PS}^k$ ,  $\Pi_{CL}^k$  and  $\Pi_{VW}^k$ , equivalence-invariance follows from the fact that all three properties are covered-edge invariant, and hereditary follows because the corresponding measures cannot increase when we remove nodes and edges from a DAG. Although we can establish EIH for the treewidth property  $\Pi_{TW}^k$  with more work, we omit further consideration of treewidth for the sake of space.

### 4.3 GENERATING DELETIONS

In this section, we show how to generate a set of deletion operators for SBES such that all  $\Pi$ -consistent deletion operators are included, for any  $\Pi \in \{\Pi_{PS}^k, \Pi_{CL}^k, \Pi_{VW}^k\}$ . Furthermore, the total number of deletion operators we generate is polynomial in the number of nodes in the domain and exponential in  $k$ .

Our approach is to restrict the  $Delete(X, Y, \mathbf{H})$  operators based on the  $\mathbf{H}$  sets and the resulting CPDAG  $\mathcal{C}'$ . In par-

ticular, we *rule out* candidate  $\mathbf{H}$  sets for which  $\Pi$  does not hold on the induced subgraph  $\mathcal{C}'[\mathbf{H} \cup X \cup Y]$ ; because all nodes in  $\mathbf{H}$  will be common children of  $X$  and  $Y$  in  $\mathcal{C}'$ —and thus a subset of the common descendants of  $X$  and  $Y$ —we know from Definition 1 (and the fact that  $\Pi$  is hereditary) that none of the dropped operators can be  $\Pi$ -consistent.

Before presenting our restricted-enumeration algorithm, we now discuss how to enumerate delete operators without restrictions. As shown by Andersson et al. (1997), a CPDAG is a chain graph whose undirected components are chordal. This means that the induced sub-graph defined over  $\mathbf{NA}_{Y,X}$ —which is a subset of the neighbors of  $Y$ —is an undirected chordal graph. A useful property of chordal graphs is that we can identify, in polynomial time, a set of *maximal cliques* over these nodes<sup>5</sup>; let  $\mathbf{C}_1, \dots, \mathbf{C}_m$  denote the nodes contained within these  $m$  maximal cliques, and let  $\bar{\mathbf{H}} = \mathbf{NA}_{Y,X} \setminus \mathbf{H}$  be the complement of the shared neighbors with respect to the candidate  $\mathbf{H}$ . Recall from Figure 2 that the preconditions for any  $Delete(X, Y, \mathbf{H})$  include the requirement that  $\bar{\mathbf{H}}$  is a clique. This means that for any valid  $\mathbf{H}$ , there must be some maximal clique  $\mathbf{C}_i$  that contains the entirety of  $\bar{\mathbf{H}}$ ; thus, we can generate all operators (without regard to any property) by stepping through each maximal clique  $\mathbf{C}_i$  in turn, initializing  $\mathbf{H}$  to be all nodes *not* in  $\mathbf{C}_i$ , and then generating a new operator corresponding to expanding  $\mathbf{H}$  by all subsets of nodes in  $\mathbf{C}_i$ . Note that if  $\mathbf{NA}_{Y,X}$  is itself a clique, we are enumerating over all  $2^{|\mathbf{NA}_{Y,X}|}$  operators.

As we show below, all three of the properties of interest impose a bound on the maximum clique size among nodes in  $\bar{\mathbf{H}}$ . If we are given such a bound  $s$ , we know that any “expansion” subset for a clique that has size greater than  $s$  will result in an operator that is not valid. Thus, we can implement the above operator-enumeration approach more efficiently by only generating subsets within each clique that have size at most  $s$ . This allows us to process each clique  $\mathbf{C}_i$  with only  $O(|\mathbf{C}_i + 1|^s)$  calls to the scoring function. In addition, we need not enumerate over *any* of the subsets of  $\mathbf{C}_i$  if, after removing this clique from the graph, there remains a clique of size greater than  $s$ ; we define the function  $FilterCliques(\{\mathbf{C}_1, \dots, \mathbf{C}_m\}, s)$  to be the subset of cliques that remain after imposing this constraint. With this function, we can define SELECTIVE-GENERATE-OPS as shown in Figure 6 to leverage the max-clique-size constraint when generating operators; this algorithm will in turn be used to generate all of the CPDAG operators during SBES.

**Example:** In Figure 7, we show an example CPDAG  $\mathcal{C}$  for which to run  $SELECTIVE-GENERATE-OPS(\mathcal{C}, X, Y, s)$  for various values of  $s$ . In the example, there is a single clique

<sup>5</sup>Blair and Peyton (1993) provide a good survey on chordal graphs and detail how to identify the maximal cliques while running maximum-cardinality search.

---

Algorithm SELECTIVE-GENERATE-OPS( $\mathcal{C}, X, Y, s$ )

---

**Input** : CPDAG  $\mathcal{C}$  with adjacent  $X, Y$  and limit  $s$

**Output**:  $\text{Ops} = \{\mathbf{H}_1, \dots, \mathbf{H}_m\}$

```

 $\text{Ops} \leftarrow \emptyset$ 
Generate maximal cliques  $\mathbf{C}_1, \dots, \mathbf{C}_m$  from  $\text{NA}_{Y,X}$ 
 $\mathbf{S} \leftarrow \text{FilterCliques}(\{\mathbf{C}_1, \dots, \mathbf{C}_m\}, s)$ 
foreach  $\mathbf{C}_i \in \mathbf{S}$  do
     $\mathbf{H}_0 \leftarrow \text{NA}_{Y,X} \setminus \mathbf{C}_i$ 
    foreach  $\mathbf{C} \subseteq \mathbf{C}_i$  with  $|\mathbf{C}| \leq s$  do
        Add  $\mathbf{H}_0 \cup \mathbf{C}$  to  $\text{Ops}$ 
    end
end
return  $\text{Ops}$ 

```

---

Figure 6: Algorithm to generate clique-size limited delete operators.

$\mathbf{C} = \{A, B\}$  in the set  $\text{NA}_{Y,X}$ , and thus at the top of the outer **foreach** loop, the set  $\mathbf{H}_0$  is initialized to the empty set. If  $s = 0$ , the only subset of  $\mathbf{C}$  with size zero is the empty set, and so that is added to  $\text{Ops}$  and the algorithm returns. If  $s = 1$  we add, in addition to the empty set, all singleton subsets of  $\mathbf{C}$ . For  $s \geq 2$ , we add all subsets of  $\mathbf{C}$ .  $\square$

Now we discuss how each of the three properties impose a constraint  $s$  on the maximum clique among nodes in  $\mathbf{H}$ , and consequently the selective-generation algorithm in Figure 6 can be used with each one, given an appropriate bound  $s$ . For both  $\Pi_{VW}^k$  and  $\Pi_{CL}^k$ , the  $k$  given imposes an explicit bound on  $s$  (i.e.,  $s = k$  for both). Because any clique in  $\mathbf{H}$  of size  $r$  will result in a DAG member of the resulting equivalence class having a node in that clique with at least  $r + 1$  parents (i.e.,  $r - 1$  from the other nodes in the clique, plus both  $X$  and  $Y$ ), we have for  $\Pi_{PS}^k$ ,  $s = k - 1$ .

We summarize the discussion above in the following proposition.

**Proposition 1.** *Algorithm SELECTIVE-GENERATE-OPS applied to all edges using clique-size bound  $s$  generates all  $\Pi$ -consistent delete operators for  $\Pi \in \{\Pi_{PS}^{s+1}, \Pi_{CL}^s, \Pi_{VW}^s\}$ .*

We now argue that running SBES on a domain of  $n$  variables when using Algorithm SELECTIVE-GENERATE-OPS with a bound  $s$  requires only a polynomial number in  $n$  of calls to the scoring function. Each clique in the inner loop of the algorithm can contain at most  $n$  nodes, and therefore we generate and score at most  $(n + 1)^s$  operators, requiring at most  $2(n + 1)^s$  calls to the scoring function. Because the cliques are maximal, there can be at most  $n$  of them considered in the outer loop. Because there are never more than  $n^2$  edges in a CPDAG, and we will delete at most all of them, we conclude that even if we decided to rescore ev-

ery operator after every edge deletion, we will only make a polynomial number of calls to the scoring function.

From the above discussion and the fact that SBES completes using at most a polynomial number of calls to the scoring function, we get the following result for the full SGEN algorithm.

**Proposition 2.** *The SGEN algorithm, when run over a domain of  $n$  variables and given  $\Pi \in \{\Pi_{PS}^{s+1}, \Pi_{CL}^s, \Pi_{VW}^s\}$ , runs to completion using a number of calls to the DAG scoring function that is polynomial in  $n$  and exponential in  $s$ .*

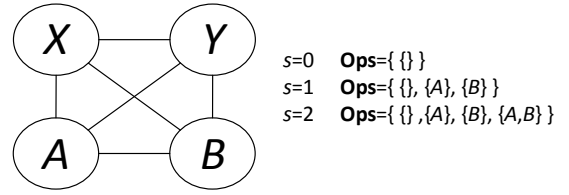


Figure 7: An example CPDAG  $\mathcal{C}$  and the resulting operators generated by SELECTIVE-GENERATE-OPS( $\mathcal{C}, X, Y, s$ ) for various values of  $s$ .

## 5 EXPERIMENTS

In this section, we present a simple synthetic experiment comparing SBES and BES that demonstrates the value of pruning operators. In our experiment we used an *oracle* scoring function. In particular, given a generative model  $\mathcal{G}$ , our scoring function computes the minimum-description-length score assuming a data size of five billion records, but without actually sampling any data: instead, we use exact inference in  $\mathcal{G}$  (i.e., instead of counting from data) to compute the conditional probabilities needed to compute the expected log loss. This allows us to get near-asymptotic behavior without the need to sample data. To evaluate the cost of running each algorithm, we counted the number of times the scoring function was called on a unique node and parent-set combination; we cached these scores away so that if they were needed multiple times during a run of the algorithm, they were only computed (and counted) once.

In Figure 8, we show the average number of scoring-function calls required to complete BES and SBES when starting from a complete graph over a domain of  $n$  binary variables, for varying values of  $n$ . Each average is taken over ten trials, corresponding to ten random generative models. We generated the structure of each generative model as follows. First, we ordered all node pairs by randomly permuting the nodes and taking each node in turn with each of its predecessors in turn. For each node pair, we chose to attempt an edge insertion with probability one half. For each attempt, we added an edge if doing so (1)

did not create a cycle and (2) did not result in a node having more than two parents; if an edge could be added in either direction, we chose the direction at random. We sampled the conditional distributions for each node and each parent configuration from a uniform Dirichlet distribution with equivalent-sample size of one. We ran SBES with  $\Pi_{PS}^2$ .

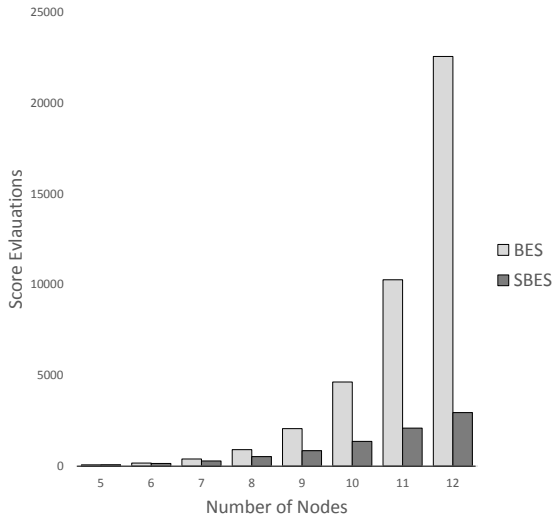


Figure 8: Number of score evaluations needed to run BES and SBES, starting from the complete graph, for a range of domain sizes.

Our results show clearly the exponential dependence of BES on the number of nodes in the clique, and the increasing savings we get with SBES by leveraging the fact that  $\Pi_{PS}^2$  holds in the generative structure.

Note that to realize large savings in practice, when GES runs FES instead of starting from a dense graph, a (relatively sparse) generative distribution must lead FES to an equivalence class containing a (relatively dense) undirected clique that is subsequently “thinned” during BES. We can synthesize challenging grid distributions to force FES into such states, but it is not clear how realistic such distributions are in practice. When we re-run the clique experiment above, but where we instead start both BES and SBES from the model that results from running FES (i.e., with no polynomial-time guarantee), the savings from SBES are small due to the fact that the subsequent equivalence classes do not contain large cliques.

## 6 CONCLUSION

We introduced a restricted version of the GES algorithm, SGES, that leverages graph-theoretic complexity properties to prune the backward-phase operators. We showed that for a particular class of properties—which includes maximum-clique size, maximum number of parents, and  $v$ -width—we can guarantee that the number of score evaluations is polynomial in the number of nodes in the domain.

The fact that we can use our approach to selectively choose operators for any hereditary and equivalence invariant graph-theoretic property provides the opportunity to explore alternative complexity measures. Another candidate complexity measure is the maximum number of  $v$ -structures. Although the corresponding property does not limit the maximum size of a *clique* in  $\mathbf{H}$ , it limits directly the size  $|\mathbf{H}|$  for every operator and thus it is easy to enumerate these operators efficiently. Another complexity measure of interest is *treewidth*, due to the fact that exact inference in a Bayesian-network model is takes time exponential in this measure.

The results we have presented are for the general Bayesian-network learning problem. It is interesting to consider the implications of our results for the problem of learning particular subsets of Bayesian networks. One natural class that we discussed in Section 2 is that of polytrees. If we assume that the generative distribution is perfect with respect to a polytree then we know the  $v$ -width of the generative graph is one. This implies, in the limit of large data, that we can recover the structure of the generative graph with a polynomial number of score evaluations. This provides a score-based recovery algorithm analogous to the constraint-based approach of Geiger et al. (1990).

We presented a simple complexity analysis for the purpose of demonstrating that SGES uses a only polynomial number of calls to the scoring function. We leave as future work a more careful analysis that establishes useful constants in this polynomial. In particular, we can derive tighter bounds on the total number of node-and-parent-configurations that are needed to score all the operators for each CPDAG, and by caching these configuration scores we can further take advantage of the fact that most operators remain valid (i.e., the preconditions still hold) and have the same score after each transformation.

Finally, we plan to investigate practical implementations of poly-FES that have the polynomial-time guarantees needed for SGES.

## References

- [1] Pieter Abbeel, Daphne Koller, and Andrew Y. Ng. Learning factor graphs in polynomial time and sample complexity. *JMLR*, 7:1743–1788, 2006.
- [2] Steen A. Andersson, David Madigan, and Michael D. Perlman. A characterization of Markov equivalence classes for acyclic digraphs. *Annals of Statistics*, 25:505–541, 1997.
- [3] Jean R. S. Blair and Barry W. Peyton. An introduction to chordal graphs and clique trees. In *Graph Theory and Sparse Matrix Computations*, pages 1–29, 1993.
- [4] David Maxwell Chickering. A transformational characterization of Bayesian network structures. In *UAI’95*, pages 87–98. 1995.
- [5] David Maxwell Chickering. Learning Bayesian networks is NP-complete. In *Learning from Data: Artificial Intelligence and Statistics V*, pages 121–130. Springer-Verlag, 1996.
- [6] David Maxwell Chickering. Optimal structure identification with greedy search. *JMLR*, 3:507–554, November 2002.
- [7] David Maxwell Chickering and Christopher Meek. Finding optimal Bayesian networks. In *UAI’02*, pages 94–102. 2002.
- [8] David Maxwell Chickering and Christopher Meek. Selective greedy equivalence search: Finding optimal Bayesian networks using a polynomial number of score evaluations. MSR-TR-2015-45, 2015.
- [9] David Maxwell Chickering, Christopher Meek, and David Heckerman. Large-sample learning of Bayesian networks is NP-hard. *JMLR*, 5:1287–1330, October 2004.
- [10] C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14:462–467, 1968.
- [11] Sanjoy Dasgupta. Learning polytrees. In *UAI’99*, pages 131–141. 1999.
- [12] Jack Edmonds. Optimum branching. *J. Res. NBS*, 71B:233–240, 1967.
- [13] Nir Friedman, Iftach Nachman, and Dana Peer. Learning Bayesian network structure from massive datasets: The “sparse candidate” algorithm. In *UAI’99*. 1999.
- [14] Serge Gaspers, Mikko Koivisto, Mathieu Liedloff, Sebastian Ordyniak, and Stefan Szeider. On finding optimal polytrees. In *AAAI’12*. 2012.
- [15] Dan Geiger, Azaria Paz, and Judea Pearl. Learning causal trees from dependence information. In *AAAI’90*, pages 770–776. AAAI Press, 1990.
- [16] Steven B. Gillispie and Michael D. Perlman. Enumerating Markov equivalence classes of acyclic digraph models. In *UAI’01*, pages 171–177. 2001.
- [17] Markus Kalisch and Peter Buhlmann. Estimating high-dimensional directed acyclic graphs with the PC algorithm. *JMLR*, 8:613–636, 2007.
- [18] David Karger and Nathan Srebro. Learning Markov networks: Maximum bounded tree-width graphs. In *SODA’01*, pages 391–401, January 2001.
- [19] Mikko Koivisto and Kismat Sood. Exact Bayesian structure discovery in Bayesian networks. *JMLR*, 5:549–573, December 2004.
- [20] Kaname Kojima, Eric Perrier, Seiya Imoto, and Satoru Miyano. Optimal search on clustered structural constraint for learning Bayesian network structure. *JMLR*, 11:285–310, 2010.
- [21] Christopher Meek. Finding a path is harder than finding a tree. *JAIR*, 15:383–389, 2001.
- [22] Mukund Narasimhan and Jeff Bilmes. PAC-learning bounded tree-width graphical models. In *UAI’04*, *UAI’04*, pages 410–417, 2004.
- [23] Sebastian Ordyniak and Stefan Szeider. Parameterized complexity results for exact Bayesian network structure learning. *JAIR*, 46:263–302, 2013.
- [24] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.
- [25] Dafna Shahaf, Anton Chechetka, and Carlos Guestrin. Learning thin junction trees via graph cuts. In *AISTATS’09*, 2009.
- [26] Tomi Silander and Petri Myllymäki. A simple approach for finding the globally optimal Bayesian network structure. In *UAI’06*, pages 445–452, 2006.
- [27] Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search*. Springer-Verlag, New York, 1993.
- [28] Ioannis Tsamardinos, Laura E. Brown, and Constantin F. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 2006.
- [29] Thomas Verma and Judea Pearl. Equivalence and synthesis of causal models. In *UAI’91*, pages 220–227, 1991.

---

# Stable Spectral Learning Based on Schur Decomposition

---

**Nicolò Colombo**

Luxembourg Centre for Systems Biomedicine  
University of Luxembourg  
nicolo.colombo@uni.lu

**Nikos Vlassis**

Adobe Research  
San Jose, CA  
vlassis@adobe.com

## Abstract

Spectral methods are a powerful tool for inferring the parameters of certain classes of probability distributions by means of standard eigenvalue-eigenvector decompositions. Spectral algorithms can be orders of magnitude faster than log-likelihood based and related iterative methods, and, thanks to the uniqueness of the spectral decomposition, they enjoy global optimality guarantees. In practice, however, the applicability of spectral methods is limited due to their sensitivity to model misspecification, which can cause instability issues in the case of non-exact models. We present a new spectral approach that is based on the Schur triangularization of an observable matrix, and we carry out the corresponding theoretical analysis. Our main result is a bound on the estimation error that is shown to depend linearly on the condition number of the ground-truth conditional probability matrix and inversely on the eigengap of an observable matrix. Numerical experiments show that the proposed method is more stable, and performs better in general, than the classical spectral approach using direct matrix diagonalization.

## 1 INTRODUCTION

The problem of learning mixtures of probability distributions from sampled data is central in the statistical literature (Titterton, 1985; Lindsay, 1995). In pioneering work, Chang (1996) showed that it is possible to learn a mixture of product distributions via the spectral decomposition of ‘observable’ matrices, that is, matrices that can be estimated directly from the data using suitable combinations of the empirical joint probability distributions (Chang, 1996). Extensions and improvements of this idea have been developed more recently in a series of works, where the spectral technique is applied to a larger class of probability distribu-

tions, including Gaussian mixtures, Hidden Markov models, stochastic languages, and others (Mossel and Roch, 2006; Hsu et al., 2012; Anandkumar et al., 2012a,c; Balle et al., 2014; Kuleshov et al., 2015). Some of the most widely studied algorithms include Chang’s spectral technique (Chang, 1996; Mossel and Roch, 2006), a tensor decomposition approach (Anandkumar et al., 2012a), and an indirect learning method for inferring the parameters of Hidden Markov Models (Hsu et al., 2012).

Spectral algorithms are typically much faster than iterative solvers such as the EM algorithm (Dempster et al., 1977), and thanks to the uniqueness of the spectral decomposition, they enjoy strong optimality guarantees. However, spectral algorithms are more sensitive to model misspecification than algorithms that maximize log-likelihood. Studies in the field of linear system subspace identification have shown that the solutions obtained via matrix decomposition methods can be suboptimal (Favoreel et al., 2000; Buesing et al., 2012). On the other hand, good results have been obtained by using the output of a spectral algorithm to initialize the EM algorithm (Zhang et al., 2014).

The practical implementation of the spectral idea is a non-trivial task because the stability of spectral decomposition strongly depends on the spacing between the eigenvalues of the empirical matrices (Anandkumar et al., 2012a; Hsu and Kakade, 2012). Mossel and Roch (2006) obtain certain eigenvalue separation guarantees for Chang’s spectral technique via the contraction of higher (order three) moments through Gaussian random vectors. Anandkumar et al. (2012a) describe a tensor decomposition method that generalizes deflation methods for matrix diagonalization to the case of symmetric tensors of order three. Another algorithmic variant involves replacing the random contracting vector of Chang’s spectral technique with an ‘anchor observation’, which guarantees the presence of at least one well separated eigenvalue (Arora et al., 2012; Song and Chen, 2014) (See also Kuleshov et al. (2015) for a similar idea). Finally, Zou et al. (2013) have presented a technique for learning mixtures of product distributions in the presence of a background model.

In this article we propose an alternative and more stable approach to Chang’s method that is based on Schur decomposition (Konstantinov et al., 1994). We show that an approximate triangularization of all observables matrices appearing in Chang’s spectral method can be obtained by means of the orthogonal matrices appearing in the Schur decomposition of their linear combination, an idea that has been suggested earlier (Corless et al., 1997; Anandkumar et al., 2012a). Our main result is a theoretical bound on the estimation error that is based on a perturbation analysis of Schur decomposition (Konstantinov et al., 1994). In analogy to related results in the literature, the bound is shown to depend directly on the model misspecification error and inversely on an eigenvalue separation gap. However, the major advantage of the Schur approach is that the bound depends very mildly on the condition number of the ground-truth conditional probability matrix (see discussion after Theorem 1). We compare numerically the proposed Schur decomposition approach with the standard spectral technique (Chang, 1996; Mossel and Roch, 2006), and we show that the proposed method is more stable and does a better job in recovering the parameters of misspecified mixtures of product distributions.

## 2 SPECTRAL LEARNING VIA SCHUR DECOMPOSITION

Here we discuss the standard spectral technique (Chang, 1996; Mossel and Roch, 2006), and the proposed Schur decomposition, in the context of learning mixtures of product distributions. The complete algorithm is shown in Algorithm 1. Its main difference to previous algorithms is step 13 (Schur decomposition).

**The spectral approach in a nutshell.** Consider  $\ell$  distinct variables taking values in a discrete set with finite number of elements  $\{1, \dots, d\}$ , and a sample  $S$  consisting of a number of independent joint observations. The empirical distribution corresponding to these observations is computed by counting the frequencies of all possible joint events in the sample (step 3), and it is modeled (approximated) by a mixture of product distributions with a given number  $p$  of mixture components. For every  $p < d$ , spectral methods allow one to recover the parameters of this approximation by means of the simultaneous diagonalisation of a set of ‘observable’ nearly diagonalizable matrices  $\{\hat{M}_1, \dots, \hat{M}_p\}$ , computed from the sample  $S$  (step 10).

If the sample is drawn exactly from a mixture of  $p$  components, and in the limit of an infinite amount of data, the mixture parameters, i.e., the conditional probability distributions and the mixing weights of the mixture, are contained exactly in the eigenvalues of the matrices  $\hat{M}_i$  (Chang, 1996). If the sample is not drawn exactly from a mixture of  $p$  product distributions, and in the typical finite

---

### Algorithm 1 Spectral algorithm via Schur decomposition

---

**Input:** data  $s_n = [x_n, y_n, z_n] \in \mathcal{N}$ , dimension  $d$ , number of mixture components  $p$

**Output:** estimated conditional probability matrices  $\hat{X}, \hat{Y}, \hat{Z}$  and mixing weights vector  $\hat{w}$

- 1:  $\hat{P} = 0$
  - 2: **for**  $s_n \in S$  **do**
  - 3:  $\hat{P}_{x_n y_n z_n} = \hat{P}_{x_n y_n z_n} + 1$
  - 4: **end for**
  - 5: **for**  $i = 1, \dots, d$  **do**
  - 6:  $[\hat{P}_i^Y]_{jk} = \hat{P}_{jik}, [\hat{P}_i^X]_{jk} = \hat{P}_{ijk}, [\hat{P}_i^Z]_{jk} = \hat{P}_{jki}$
  - 7: **end for**
  - 8: compute  $[\hat{P}_{xz}] = \sum_i [\hat{P}_i^Y], [\hat{P}_{yz}] = \sum_i [\hat{P}_i^X], [\hat{P}_{xy}] = \sum_i [\hat{P}_i^Z]$
  - 9: **for**  $i = 1, \dots, d$  **do**
  - 10: compute  $\hat{M}_i = \hat{P}_i^Y \hat{P}_{xz}^{-1}$
  - 11: **end for**
  - 12: find  $\theta \in \mathbf{R}^d$  such that  $\hat{M} = \sum_i \theta_i \hat{M}_i$  has real non-degenerate eigenvalues.
  - 13: find  $\hat{U}$  such that  $\hat{U}^T \hat{U} = 1$  and  $\hat{U}^T \hat{M} \hat{U}$  is upper triangular (Schur decomposition)
  - 14: **for**  $i, j = 1, \dots, d$  **do**
  - 15: let  $\hat{Y}_{i,j} = [\hat{U}^T \hat{M}_i \hat{U}]_{jj}$
  - 16: set  $\hat{Y}_{i,j} = 0$  if  $[\hat{U}^T \hat{M}_i \hat{U}]_{jj} < 0$
  - 17: **end for**
  - 18: normalize to 1 the columns of  $\hat{Y}$
  - 19: **for**  $i = 1, \dots, d$  **do**
  - 20: compute  $\hat{M}_i^X = \hat{P}_i^X \hat{P}_{yz}^{-1}$
  - 21: compute  $\hat{M}_i^Z = \hat{P}_i^Z \hat{P}_{xy}^{-1}$
  - 22: **end for**
  - 23: **for**  $i, j = 1, \dots, d$  **do**
  - 24: let  $\hat{X}_{i,j} = [\hat{Y}^{-1} \hat{M}_i^X \hat{Y}]_{jj}$  and set  $\hat{X}_{i,j} = 0$  if  $[\hat{Y}^{-1} \hat{M}_i^X \hat{Y}]_{jj} < 0$
  - 25: let  $\hat{Z}_{i,j} = [\hat{X}^{-1} \hat{M}_i^Z \hat{X}]_{jj}$  and set  $\hat{Z}_{i,j} = 0$  if  $[\hat{X}^{-1} \hat{M}_i^Z \hat{X}]_{jj} < 0$
  - 26: **end for**
  - 27: normalize to 1 the columns of  $\hat{X}$  and  $\hat{Z}$
  - 28: compute  $\hat{w} = \hat{X}^{-1} \hat{P}_{xy} (\hat{Y}^T)^{-1}$  and normalize to 1
-

sample setting, the model is only an approximation to the empirical distribution, and as a result, the matrices  $\hat{M}_i$  are no longer simultaneously diagonalizable and an approximate diagonalisation technique is required. The standard approach consists of choosing one particular observable matrix in the set, or a linear combination of all matrices, and use its eigenvectors to diagonalize each  $\hat{M}_i$  (Mossel and Roch, 2006).

Here we propose a new approach that is based on the Schur decomposition of a linear combination of the observable matrices. In particular, we first mix the matrices  $\hat{M}_i$  to compute a candidate matrix  $\hat{M}$  (step 12), and then we apply Schur decomposition to  $\hat{M}$  (step 13). The eigenvalues of each  $\hat{M}_i$ , and thereby the model parameters, are then extracted using the orthogonal matrix  $\hat{U}$  of the Schur decomposition (steps 15-16). Effectively we exploit the fact that the real eigenvalues of a matrix  $A$  always appear on the diagonal of its Schur triangularization  $T = U^T A U$ , even though the entries of the strictly upper diagonal part of  $T$  may not be unique. Using the perturbation analysis of the Schur system of a matrix by Konstantinov et al. (1994), we obtain a theoretical bound on the error of such eigenvalue estimation as a function of the model misspecification error, the condition number of the ground-truth matrix  $X$ , and the separation of the eigenvalues of  $\hat{M}$  (Theorem 1).

**Detailed description and the Schur approach.** Consider for simplicity a sample  $S$  of independent observations  $s = [x, y, z]$  of three distinct variables taking values in the discrete set  $\{1, \dots, d\}$ . The empirical distribution associated to the sample  $S$  is defined as

$$\hat{P}_{i,j,k} = \frac{1}{|S|} \sum_{s \in S} \delta_{x,i} \delta_{y,j} \delta_{z,k} \quad (1)$$

where  $|S|$  is the number of elements in  $S$  and  $\delta_{ab} = 1$  if  $a = b$  and zero otherwise. The empirical distribution  $\hat{P}$  is a nonnegative order-3 tensor whose entries sum to one. Its nonnegative rank  $\text{rank}_+(\hat{P})$  is the minimum number of rank-1 tensors, i.e., mixture components, required to express  $\hat{P}$  as a mixture of product distributions. Such a decomposition of  $\hat{P}$  (exact or approximate) is always possible (Lim and Comon, 2009). Hence, for any choice of  $p \leq \text{rank}_+(\hat{P})$ , we can hypothesize that  $\hat{P}$  is generated by a model

$$\hat{P} = P + \epsilon \Delta P, \quad \epsilon \geq 0 \quad (2)$$

where  $P \in [0, 1]^{d_x \times d_y \times d_z}$  is a nonnegative rank- $p$  approximation of  $\hat{P}$ ,  $\epsilon$  is a model misspecification parameter, and  $\Delta P \in [0, 1]^{d_x \times d_y \times d_z}$  is a nonnegative tensor whose entries sum to one. The rank- $p$  component  $P$  is interpreted as the mixture of product distributions that approximates the empirical distribution, and it can be written

$$P_{ijk} = \sum_{h=1}^p w_h X_{ih} Y_{jh} Z_{kh}, \quad (3)$$

where  $w_h \in [0, 1]$  for all  $h = 1, \dots, p$ , and we have defined the conditional probability matrices

$$X \in [0, 1]^{d \times p}, \quad \mathbf{1}_d^T X = \mathbf{1}_p^T, \quad (4)$$

(and similarly for  $Y, Z$ ), where  $\mathbf{1}_n$  is a vector of  $n$  ones. The columns of the matrices  $X, Y, Z$  encode the conditional probabilities associated with the  $p$  mixture components, and the mixing weights satisfy  $\sum_h w_h + \epsilon = 1$ .

The conditional probability matrices  $X, Y, Z$  and the mixing weight  $w$  of the rank- $p$  mixture can be estimated from the approximate eigenvalues of a set of observable matrices  $\hat{M}_i$ , for  $i = 1, \dots, p$ , that are computed as follows. Let for simplicity  $p = d$  and consider the matrices  $[\hat{P}_i^Y]_{jk} = \hat{P}_{jik}$  (step 6) and  $[\hat{P}_{xz}] = \sum_i [\hat{P}_i^Y]$  (step 8). Assuming that  $\hat{P}_{xz}$  is invertible, we define (step 10)

$$\hat{M}_i = \hat{P}_i^Y \hat{P}_{xz}^{-1} \quad (5)$$

for  $i = 1, \dots, d$ . Under the model assumption  $\hat{P} = P + \epsilon \Delta P$ , it is easy to show that

$$\hat{M}_i = M_i + \Delta M_i + o(\epsilon^2) \quad (6)$$

where  $\Delta M_i \in \mathbf{R}^{d \times d}$  is linear in the misspecification parameter  $\epsilon$ , and

$$M_i = X \text{diag}(Y_{i1}, \dots, Y_{ip}) X^{-1} \quad (7)$$

where  $\text{diag}(v_1, \dots, v_d)$  denotes a diagonal matrix whose diagonal entries are  $v_1, \dots, v_d$ . If the model is exact, i.e.,  $p = \text{rank}_+(\hat{P})$  or equivalently  $\epsilon = 0$ , the matrices  $\{\hat{M}_i\}$  are simultaneously diagonalizable and the entries of the conditional probability matrix  $Y$  are given (up to normalization of its columns) by

$$Y_{ij} \propto [V^{-1} \hat{M}_i V]_{jj}, \quad i, j = 1, \dots, d \quad (8)$$

where  $V$  is the matrix of the eigenvectors shared by all  $\hat{M}_i$ .

When  $\epsilon \neq 0$ , the matrices  $\hat{M}_i$  are no longer simultaneously diagonalizable and an approximate simultaneous diagonalisation scheme is needed. The standard procedure consists of selecting a representative matrix  $\hat{M}$ , compute its eigenvectors  $\hat{V}$ , and use the matrix  $\hat{V}$  to obtain the approximate eigenvalues of all matrices  $\hat{M}_i$  and thereby estimate  $Y_{ij}$  (Mossel and Roch, 2006; Hsu et al., 2012; Anandkumar et al., 2012b). In this case, the estimation error is known to depend on the model misspecification parameter  $\epsilon$  and on the inverse of an eigenvalue separation  $\gamma$  (see, e.g., eq. (12)). Using matrix perturbation theorems and properties of the Gaussian distribution, Mossel and Roch (2006) have shown that a certain separation  $\gamma > \alpha$  is guaranteed with probability proportional to  $(1 - \alpha)$  if  $\hat{V}$  is the matrix of the eigenvectors of some  $\hat{M} = \sum_i \theta_i \hat{M}_i$ , with  $\theta$  sampled from a Gaussian distribution of zero mean and unit variance. In practice, however, this approach can give rise to



instabilities (such as negative or imaginary values for  $Y_{ij}$ ), especially when the size of the empirical matrices grows.

Here we propose instead to triangularize the matrices  $\hat{M}_i$  by means of the Schur decomposition of their linear combination  $\hat{M} = \sum_i \theta_i \hat{M}_i$ , for an appropriate  $\theta$  (steps 12-13). The orthogonal matrix  $\hat{U}$  obtained from the Schur decomposition  $\hat{M} = \hat{U} \hat{T} \hat{U}^T$  is then used in place of the eigenvectors matrix of  $\hat{M}$  to approximately triangularize all the observable matrices  $\hat{M}_i$  and thereby recover the mixture parameters (steps 15 and 24, 25). For example, the conditional probability matrix  $Y$  is estimated as

$$\hat{Y}_{ij} \propto [\hat{U}^T \hat{M}_i \hat{U}]_{jj}, \quad i, j = 1, \dots, d \quad (9)$$

and normalized so that its columns sum to one. Let  $\|\cdot\|$  denote the Frobenius norm. Our main result is the following:

**Theorem 1.** *Let  $\hat{M}_i$  and  $M_i$  be the real  $d \times d$  matrices defined in (5) and (6). Suppose it is possible to find  $\theta \in \mathbf{R}^d$  such that  $\hat{M} = \sum_k \theta_k \hat{M}_k$  has real distinct eigenvalues. Then, for all  $j = 1, \dots, d$ , there exists a permutation  $\pi$  such that*

$$|\hat{Y}_{ij} - Y_{i\pi(j)}| \leq \left( a_1 \frac{k(X) \lambda_{\max}}{\hat{\gamma}} + 1 \right) E + o(E^2) \quad (10)$$

where  $k(X) = \frac{\sigma_{\max}(X)}{\sigma_{\min}(X)}$  is the condition number of the ground-truth conditional probability matrix  $X$ ,  $\lambda_{\max} = \max_{i,j} Y_{i,j}$ ,

$$\hat{\gamma} = \min_{i \neq j} |\lambda_i(\hat{M}) - \lambda_j(\hat{M})| > 0 \quad (11)$$

with  $\lambda_i(\hat{M})$  being the  $i$ th eigenvalue of  $\hat{M}$ ,  $a_1 = \|\theta\| \sqrt{\frac{2^3 d^2}{d-1}}$ , and  $E = \max_i \|\Delta M_i\| = \max_i \|\hat{M}_i - M_i\|$ .

*Proof.* See appendix.  $\square$

The analogous bound for the diagonalization approach is (Anandkumar et al., 2012c, Section B6, eq. 11)

$$|\hat{Y}_{ij} - Y_{i\pi(j)}| \leq \left( a_2 k(X)^4 \frac{\tilde{\lambda}_{\max}}{\gamma} + a_3 k(X)^2 \right) E, \quad (12)$$

where  $\gamma = \min_{i \neq j} |\lambda_i(\sum_k \theta_k M_k) - \lambda_j(\sum_k \theta_k M_k)|$ ,  $\tilde{\lambda}_{\max} = \max(\max_i [\theta^T Y]_i, \max_{i,j} Y_{i,j})$ , and  $a_2, a_3$  are constants that depend on the dimensions of the involved matrices.

When the model misspecification error  $E$  is not too large, the error bound under the Schur approach (10) is characterized by a much smoother dependence on  $k(X)$  than the error bound (12). Moreover, the Schur bound depends on the eigenvalue gap  $\hat{\gamma}$  of an observable matrix, and hence it can be controlled in practice by optimizing  $\theta$ . The simplified dependence on  $k(X)$  of the Schur bound is due to

the good perturbation properties of the orthogonal matrices involved in the Schur decomposition, as compared to the eigenvector matrices of the Chang approach. The difference in the bounds suggests that, for a randomly generated true model, a spectral algorithm based on the Schur decomposition is expected to be more stable and accurate in practice than an algorithm based on matrix diagonalization. Intuitively, the key to the improved stability of the Schur approach comes from the freedom to ignore the non-unique off-diagonal parts in Schur triangulation.

During the reviewing process we were made aware of the work of Kuleshov et al. (2015), who propose computing a tensor factorization from the simultaneous diagonalization of a set of matrices obtained by projections of the tensor along random directions. Kuleshov et al. (2015) establish an error bound that is independent of the eigenvalue gap, but their approach does not come with global optimality guarantees (but the authors report good results in practice). It would be of interest to see whether such random projections combined with a simultaneous Schur decomposition (see, e.g., De Lathauwer et al. (2004)) could offer improved bounds.

### 3 EXPERIMENTS

We have compared the performance of the proposed spectral algorithm based on Schur decomposition with the classical spectral method based on eigenvalue decomposition. The two algorithms that we tested are equivalent except for line 13 of Algorithm 1, which in the classical spectral approach should be “find  $V$  such that  $V^{-1} M V = D$ , with  $D$  diagonal”. In all experiments we used the same code with decompositions performed via the two Matlab functions `schur(M)` and `eig(M)` respectively. We tested the two algorithms on simulated real multi-view and Hidden Markov Model data. In what follows we denote by ‘schur’ the algorithm based on the Schur decomposition and by ‘eig’ the algorithm based on the eigenvalues-eigenvector decomposition.

In the first set of experiments we generated multi-view data from a mixture of product distributions of  $p$  mixture components in  $d$  dimensions. For each experiment, we created two different datasets  $\mathcal{N} = \{[x_n y_n z_n] \in [1, \dots, d]^3\}$ , and  $\mathcal{N}_{test}$ , one for training and one for testing, the latter containing the labels  $L \in [1, \dots, p]^{| \mathcal{N}_{test} |}$  of the mixture components that generated each instance. The output was evaluated by measuring the distance between the estimated conditional probability distributions  $\hat{X}, \hat{Y}, \hat{Z}$  and the corresponding ground-truth values  $X, Y, Z$ :

$$E = \|\hat{X} - X\|^2 + \|\hat{Y} - Y\|^2 + \|\hat{Z} - Z\|^2. \quad (13)$$

Since the order of the columns in  $\hat{X}, \hat{Y}, \hat{Z}$  may be different from  $X, Y, Z$ , the norms were computed after obtaining the best permutation. We also tested according to a

| $ \mathcal{N} (d = 10, p = 5)$ | $E_{schur}$          | $E_{eig}$      | $S_{schur}$           | $S_{eig}$            | $\ \hat{T}_{schur} - T\ $ | $\ \hat{T}_{eig} - T\ $ |
|--------------------------------|----------------------|----------------|-----------------------|----------------------|---------------------------|-------------------------|
| 1000                           | <b>0.057</b> (0.013) | 0.066 (0.0167) | <b>0.364</b> (0.135)  | 0.360 (0.135)        | <b>0.016</b> (0.004)      | 0.026 (0.009)           |
| 2000                           | <b>0.039</b> (0.008) | 0.120 (0.227)  | <b>0.415</b> (0.068)  | 0.356 (0.138)        | <b>0.011</b> (0.004)      | 0.019 (0.008)           |
| 5000                           | <b>0.043</b> (0.012) | 0.046 (0.013)  | <b>0.387</b> (0.114)  | 0.386 (0.084)        | <b>0.013</b> (0.004)      | 0.021 (0.004)           |
| 10000                          | <b>0.036</b> (0.014) | 0.047 (0.009)  | 0.390 (0.130)         | <b>0.402</b> (0.085) | <b>0.013</b> (0.006)      | 0.022 (0.007)           |
| 20000                          | <b>0.032</b> (0.014) | 0.113 (0.230)  | 0.431 (0.124)         | <b>0.341</b> (0.157) | <b>0.011</b> (0.007)      | 0.025 (0.007)           |
| 50000                          | <b>0.019</b> (0.015) | 0.025 (0.010)  | <b>0.475</b> (0.1887) | 0.434 (0.143)        | <b>0.007</b> (0.006)      | 0.015 (0.009)           |

Table 1: Columns recovery error  $E$ , classification score  $S$ , and distance of the approximate distribution  $\|\hat{T} - T\|$  for multi-view datasets of increasing size. The algorithm based on Schur decomposition obtained the best scores on almost all datasets.

classification rule where the estimated conditional probability matrices  $[\hat{X}, \hat{Y}, \hat{Z}]$  were used to assign each triple in the test dataset to one of the mixture components. For every run, we obtained a classification score by counting the number of successful predictions divided by the number of elements in the test dataset:

$$S = \frac{1}{|\mathcal{N}_{test}|} \sum_{n \in \mathcal{N}_{test}} f(n), \quad (14)$$

$$f(n) = \begin{cases} 0 & \arg \max_i \hat{X}_{x_n i} \hat{Y}_{y_n i} \hat{Z}_{z_n i} \neq L(n) \\ 1 & \arg \max_i \hat{X}_{x_n i} \hat{Y}_{y_n i} \hat{Z}_{z_n i} = L(n) \end{cases}. \quad (15)$$

Finally we computed the distance in norm between the recovered tensor

$$\hat{T}_{ijk} = \sum_r \hat{w}_r [\hat{X}]_{ir} [\hat{Y}]_{jr} [\hat{Z}]_{kr} \quad (16)$$

and the original tensor

$$T_{ijk} = \sum_r [w]_r [X]_{ir} [Y]_{jr} [Z]_{kr} \quad (17)$$

as follows

$$\|\hat{T} - T\| = \sqrt{\sum_{i,j,k} [\hat{T} - T]_{ijk}^2}. \quad (18)$$

In Table 1 we show the results obtained by the two algorithms for  $d = 10, p = 5$  and increasing size of the training dataset  $|\mathcal{N}|$ . In the table we report the average score over 10 analogous runs and the corresponding standard deviation in brackets. When the recovered matrices contained infinite values we have set  $E = \|X\|^2 + \|Y\|^2 + \|Z\|^2$ , and  $\|\hat{T} - T\| = \|T\|$ . The proposed algorithm based on Schur decomposition obtained the best scores on almost all datasets.

In the second set of experiments we tested the two algorithms on datasets generated by a  $d$ -dimensional Hidden Markov Model with  $p$  hidden states. For each experiment we randomly picked a model  $M_{true} = M_{true}(O_{true}, R_{true}, h_{true})$ , where  $O_{true} \in [0, 1]^{d \times p}$  is the observation matrix,  $R_{true} \in [0, 1]^{p \times p}$  is the transition

matrix, and  $h_{true} \in [0, 1]^p$  is the starting distribution, and we generated two sample datasets, one for training and one for testing. All sequences  $s_n$  were simulated starting from an initial hidden state drawn from  $h_{true}$  and following the dynamics of the model according to  $R_{true}$  and  $O_{true}$ . The length of the sequences in the training and testing datasets was set to 20. We evaluated the two algorithms based on the columns recovery error  $E$  as in the previous set of experiments. Also, letting  $O_{true} = [o_{true1}, \dots, o_{truep}]$  and  $O = [o_1, \dots, o_p]$ , we considered a recovery ratio  $R(M) = \frac{r}{p}$ , where  $r$  is the number of columns satisfying

$$\|o_{truei} - o_i\|^2 < \xi, \quad \xi = 0.05^2 * d. \quad (19)$$

In Table 2 we show the results for recovering a  $d = \{5, 10, 20, 30\}$  HMM with  $p = 5$  hidden states. All values are computed by averaging over 10 experiments and the corresponding standard variation is reported between brackets. The Schur algorithm is in general better than the classical approach. We note that, for a fixed number of hidden states, the inference of the HMM parameters becomes harder as the dimensionality of the space decreases. As the recovery ratio  $R$  shows, in the limit situation  $d = p$  both algorithms fail (values  $R = 0$  imply unstable solutions).

## 4 CONCLUSIONS

We have presented a new spectral algorithm for learning multi-view mixture models that is based on the Schur decomposition of an observable matrix. Our main result is a theoretical bound on the estimation error (Theorem 1), which is shown to depend very mildly (and much more smoothly than in previous results) on the condition number of the ground-truth conditional probability matrix, and inversely on the eigengap of an observable matrix. Numerical experiments show that the proposed method is more stable, and performs better in general, than the classical spectral approach using direct matrix diagonalization.

### Appendix - Proof of Theorem 1

**Theorem 1.** *Let  $\hat{M}_i$  and  $M_i$  be the real  $d \times d$  matrices defined in (5) and (6). Suppose it is possible to find  $\theta \in \mathbf{R}^d$  such that  $\hat{M} = \sum_k \theta_k M_k$  has real distinct eigenvalues.*

| $ \mathcal{N} (d=30, p=5)$ | $E(T_{schur})$       | $E(T_{eig})$         | $R(T_{schur})$       | $R(T_{eig})$         |
|----------------------------|----------------------|----------------------|----------------------|----------------------|
| 100                        | <b>0.011</b> (0.001) | 0.014 (0.005)        | 1 (0)                | 1 (0)                |
| 500                        | <b>0.011</b> (0.001) | 0.012 (0.002)        | 1 (0)                | 1 (0)                |
| 1000                       | <b>0.011</b> (0.001) | 0.013 (0.002)        | 1 (0)                | 1 (0)                |
| 2000                       | 0.011 (0.001)        | 0.011 (0.001)        | 1 (0)                | 1 (0)                |
| 5000                       | 0.010 (0.001)        | 0.010 (0.001)        | 1 (0)                | 1 (0)                |
| $ \mathcal{N} (d=20, p=5)$ | $E(T_{schur})$       | $E(T_{eig})$         | $R(T_{schur})$       | $R(T_{eig})$         |
| 100                        | <b>0.019</b> (0.002) | 0.026 (0.010)        | <b>1</b> (0)         | 0.880 (0.168)        |
| 500                        | <b>0.018</b> (0.003) | 0.044 (0.080)        | <b>1</b> (0)         | 0.900 (0.316)        |
| 1000                       | <b>0.019</b> (0.004) | 0.021 (0.002)        | 1 (0)                | 1 (0)                |
| 2000                       | 0.017 (0.002)        | 0.017 (0.002)        | 1 (0)                | 1 (0)                |
| 5000                       | <b>0.015</b> (0.002) | 0.018 (0.006)        | <b>1</b> (0)         | 0.960(0.126)         |
| $ \mathcal{N} (d=10, p=5)$ | $E(T_{schur})$       | $E(T_{eig})$         | $R(T_{schur})$       | $R(T_{eig})$         |
| 100                        | <b>0.047</b> (0.011) | 0.050 (0.011)        | 0.200 (0.188)        | <b>0.220</b> (0.175) |
| 500                        | <b>0.044</b> (0.008) | 0.097 (0.154)        | <b>0.240</b> (0.157) | 0.200 (0.188)        |
| 1000                       | <b>0.046</b> (0.017) | 0.051 (0.018)        | <b>0.260</b> (0.211) | 0.120 (0.139)        |
| 2000                       | <b>0.043</b> (0.016) | 0.048 (0.011)        | <b>0.180</b> (0.220) | 0.120 (0.139)        |
| 5000                       | <b>0.040</b> (0.013) | 0.089 (0.153)        | <b>0.380</b> (0.257) | 0.200 (0.133)        |
| $ \mathcal{N} (d=5, p=5)$  | $E(T_{schur})$       | $E(T_{eig})$         | $R(T_{schur})$       | $R(T_{eig})$         |
| 100                        | <b>0.163</b> (0.089) | 0.164 (0.074)        | 0 (0)                | 0 (0)                |
| 500                        | <b>0.173</b> (0.063) | 0.228 (0.294)        | 0 (0)                | <b>0.040</b> (0.084) |
| 1000                       | <b>0.191</b> (0.068) | 0.251 (0.273)        | 0.020 (0.063)        | <b>0.040</b> (0.084) |
| 2000                       | 0.205 (0.070)        | <b>0.166</b> (0.052) | <b>0.060</b> (0.096) | 0 (0)                |
| 5000                       | <b>0.142</b> (0.063) | 0.164 (0.069)        | 0 (0)                | <b>0.020</b> (0.063) |

Table 2: Columns recovery error  $E$  and recovery ratio  $R$  for recovering HMMs of various dimensionality and hidden states using the Schur and the standard spectral approach. See text for details.

Then, for all  $j = 1, \dots, d$ , there exists a permutation  $\pi$  such that

$$|\hat{Y}_{ij} - Y_{i\pi(j)}| \leq \left( a_1 \frac{k(X) \lambda_{\max}}{\hat{\gamma}} + 1 \right) E + o(E^2) \quad (20)$$

with  $\hat{Y}$  estimated from (9), and where  $k(X) = \frac{\sigma_{\max}(X)}{\sigma_{\min}(X)}$  is the condition number of the ground-truth conditional probability matrix  $X$ ,  $\lambda_{\max} = \max_{i,j} Y_{i,j}$ ,

$$\hat{\gamma} = \min_{i \neq j} |\lambda_i(\hat{M}) - \lambda_j(\hat{M})| > 0 \quad (21)$$

with  $\lambda_i(\hat{M})$  being the  $i$ th eigenvalue of  $\hat{M}$ ,  $a_1 = \|\theta\| \sqrt{\frac{2^3 d^2}{d-1}}$ , and  $E = \max_i \|\Delta M_i\| = \max_i \|\hat{M}_i - M_i\|$ .

*Proof.* Consider the set of real commuting matrices  $M_i$ ,  $i = 1, \dots, d$ , and their random perturbations  $\hat{M}_i = M_i + \Delta M_i$  defined in (5) and (6). Assume that  $\|\Delta M_i\| < E$  for all  $i = 1, \dots, d$  and that  $\theta \in \mathbf{R}^d$  is such that the eigenvalues of  $\hat{M} = M + \Delta M = \sum_i \theta_i (M_i + \Delta M_i)$  are real and non-degenerate. Let  $\hat{U}$  be the orthogonal matrix defined by the Schur decomposition of  $\hat{M} = \hat{U}^T \hat{T} \hat{U}$  computed by the matrix decomposition subroutine in Algorithm 1. Note that  $\hat{U}$  may not be unique and different choices of  $\hat{U}$  lead to different entries in the strictly upper-diagonal part of  $\hat{T}$ . However, for any given  $\hat{U}$  such that  $\hat{U}^T \hat{T} \hat{U}$  is upper triangular, there exists an orthogonal matrix  $U$  and a real matrix  $\Delta U \in \mathbf{R}^{d,d}$  such that  $U = \hat{U} + \Delta U$  and

$$U^T M U = (\hat{U} + \Delta U)^T (\hat{M} - \Delta M) (\hat{U} + \Delta U) \quad (22)$$

$$= \hat{T} - \Delta T \quad (23)$$

$$= T \quad (24)$$

with  $T$  upper triangular. Let  $Y_{i,j}$  be the ground-truth matrix defined in (3) and  $\hat{Y}$  the estimation output by Algorithm 1. Then, assuming that  $\Delta M$  and  $\Delta U$  are small, there exists a permutation of the indexes  $\pi$  such that, for all  $i, j = 1, \dots, d$

$$\delta_y = |\hat{Y}_{ij} - Y_{i\pi(j)}| \quad (25)$$

$$= |[\hat{U}^T \hat{M}_i \hat{U}]_{jj} - [U^T M_i U]_{\pi(j)\pi(j)}| \quad (26)$$

$$\leq \|(U - \Delta U)^T (M_i + \Delta M_i) (U - \Delta U) - T_i\| \quad (27)$$

$$= \|\Delta U^T U T_i + T_i U^T \Delta U - U^T \Delta M_i U + o(\Delta^2)\| \quad (28)$$

$$= \|x T_i - T_i x + U^T \Delta M_i U + o(\Delta^2)\| \quad (29)$$

$$\leq 2 \|x\| \|T_i\| + \|\Delta M_i\| + o(\|\Delta^2\|) \quad (30)$$

$$\leq 2 \|x\| \mu + E + o(\|\Delta^2\|) \quad (31)$$

where we have defined  $x = U^T \Delta U$ ,  $\mu = \max_i \|M_i\|$  and used  $1 = (U + \Delta U)^T (U + \Delta U) = 1 + x^T + x + o(\Delta^2)$  where  $o(\Delta^2) = o(x^2) + o(\Delta M x)$ .

Following (Konstantinov et al., 1994), a linear bound of  $\|x\|$  can be estimated as follows. First, observe that the Schur decomposition of  $M$  in (24) implies

$$\text{low}(\hat{T} \hat{x} - \hat{x} \hat{T}) = \text{low}(\hat{U}^T \Delta M \hat{U}) + o(\Delta^2) \quad (32)$$

where  $\text{low}(A)$  denotes the strictly lower diagonal part of  $A$  and  $\hat{x} = \hat{U}^T \Delta U$ . Since  $\hat{T}$  is upper triangular, one has  $\text{low}(\hat{T} \hat{x} - \hat{x} \hat{T}) = \text{low}(\hat{T} \text{low}(\hat{x}) - \text{low}(\hat{x}) \hat{T})$ , i.e. the linear operator defined by  $\mathcal{L}_{\hat{T}}(\hat{x}) = \text{low}(\hat{T} \hat{x} - \hat{x} \hat{T})$  maps strictly lower-triangular matrices to strictly lower-triangular matrices. Let  $\tilde{\mathcal{L}}_{\hat{T}}(\cdot)$  be the restriction of  $\mathcal{L}_{\hat{T}}(\cdot)$  to the subspace

of lower-triangular matrices, then from (32) one has

$$\widetilde{\mathcal{L}}_{\hat{T}}(\text{low}(\hat{x})) = \text{low}(\hat{U}^T \Delta M \hat{U}) + o(\Delta^2) \quad (33)$$

and the operator  $\widetilde{\mathcal{L}}_{\hat{T}}$  is invertible. The invertibility of  $\widetilde{\mathcal{L}}_{\hat{T}}$  follows from the non-singularity of its matrix representation  $\text{mat}(\widetilde{\mathcal{L}}_{\hat{T}})$  defined by

$$\text{vec}\left(\widetilde{\mathcal{L}}_{\hat{T}}(\text{low}(\hat{x}))\right) = \text{mat}(\widetilde{\mathcal{L}}_{\hat{T}})L \text{vec}(\text{low}(\hat{x})) \quad (34)$$

where  $\text{vec}(A)$  is the columnwise vector representation of  $A$  and  $L = [L_{ij}] \in [0, 1]^{\frac{d(d-1)}{2} \times d^2}$  the projector to the subspace of vectorized lower-triangular matrices

$$L_{ij} \in [0, 1]^{d-i \times d}, \quad i, j = 1, \dots, d-1 \quad (35)$$

$$L_{ij} = \begin{cases} 0_{d-i, d} & i \neq j \\ [0_{d-i, i}, 1_{d-i}] & i = j \end{cases} \quad (36)$$

More explicitly,  $\text{mat}(\widetilde{\mathcal{L}}_{\hat{T}}) = L(1 \otimes \hat{T} - \hat{T}^T \otimes 1)L^T$  is a block lower-triangular matrix  $\text{mat}(\widetilde{\mathcal{L}}_{\hat{T}}) = [\mathcal{M}_{ij}] \in \mathbf{R}^{\frac{d(d-1)}{2} \times \frac{d(d-1)}{2}}$  where

$$[\mathcal{M}_{ij}] \in \mathbf{R}^{d-i \times d-j}, \quad i, j = 1, \dots, d-1 \quad (37)$$

$$\mathcal{M}_{ij} = \begin{cases} [0_{d-i, i-j}, 1_{d-j}] & i > j \\ [m_i] & i = j \\ 0 & i < j \end{cases} \quad (38)$$

$$[m_i]_{jk} = \begin{cases} \hat{T}_{j+i-1, k+i} & j < k \\ \hat{T}_{j+i, j+i} - T_{i, i} & j = k \\ 0 & j > k \end{cases} \quad (39)$$

for  $i, j = 1, \dots, d-1$ . The determinant of  $\mathcal{M}$  is the product of the determinants of its diagonal blocks, *i.e.*

$$\det(M) = \prod_{i>j} (\hat{T}_{ii} - \hat{T}_{jj}) \quad (40)$$

and is not null provided that the eigenvalues of  $\hat{T}$  are real separated. In this case, the matrix  $\mathcal{M}$  and hence the operator  $\widetilde{\mathcal{L}}_{\hat{T}}(\cdot)$  are invertible. From (32) one has

$$\text{low}(\hat{x}) = \widetilde{\mathcal{L}}_{\hat{T}}^{-1} \text{low}(\hat{U}^T \Delta M \hat{U}) + o(\Delta^2) \quad (41)$$

and in particular

$$\|\hat{x}\| = \sqrt{2} \|\text{low}(\hat{x})\| \quad (42)$$

$$= \sqrt{2} \|\widetilde{\mathcal{L}}_{\hat{T}}^{-1}\|_F \|\Delta M\| + o(\|\Delta\|^2) \quad (43)$$

where the first equality is obtained using the linear approximation  $\hat{x} = -\hat{x}^T$  and  $\|A\|^2 = \|\text{low}(A)\|^2 + \|\text{diag}(A)\|^2 + \|\text{up}(A)\|^2$ , with  $\text{diag}(A)$  and  $\text{up}(A)$  denoting the diagonal and upper-diagonal parts of  $A$ . The norm of the inverse operator can be bound using its matrix realization, *i.e.*

$$\|\widetilde{\mathcal{L}}_{\hat{T}}^{-1}\|_F = \|\mathcal{M}^{-1}\| \leq \frac{1}{\sigma_{\min}(\mathcal{M})} \quad (44)$$

where  $\sigma_{\min}(A)$  is the smallest singular value of  $A$ . We can estimate  $\sigma_{\min}(\mathcal{M})$  by using the following lemma

$$\sigma_{\min}(A) = \min_{\text{rank}(B) < n} \|A - B\|, \quad \text{rank}(A) = n \quad (45)$$

and observing that the rank deficient matrix closest to  $\mathcal{M}$  is obtained by setting  $\hat{T} = T_{\text{singular}}$  in (38), where  $T_{\text{singular}}$  is defined by

$$[T_{\text{singular}}]_{i,j} = \begin{cases} \hat{T}_{j^*, j^*} & \text{if } i = j = i^* \\ \hat{T}_{i, j} & \text{otherwise} \end{cases} \quad (46)$$

with  $(i^*, j^*) = \arg \min_{i \neq j} |\hat{T}_{ii} - \hat{T}_{jj}|$ . One has

$$\sigma_{\min}(\mathcal{M}) = \|\mathcal{M} - \mathcal{M}_{\text{singular}}\| \quad (47)$$

$$= \sqrt{\sum_{i,j} (\mathcal{M} - \mathcal{M}_{\text{singular}})_{i,j}^2} \quad (48)$$

$$= \sqrt{d-1} \hat{\gamma} \quad (49)$$

$$\hat{\gamma} = |\hat{T}_{i^* i^*} - \hat{T}_{j^* j^*}| = \min_{i \neq j} |\hat{T}_{ii} - \hat{T}_{jj}| \quad (50)$$

where the last equality is obtained by noting that, for all  $i = 1, \dots, d$ , the element  $\hat{T}_{ii}$  appears  $d-1$  times in  $\mathcal{M}$ .

The norm upper bound  $\mu$  in (31) obeys

$$\mu = \max_i \|M_i\| \quad (51)$$

$$= \max_i \|X \text{diag}(Y_{i1}, \dots, Y_{id}) X^{-1}\| \quad (52)$$

$$\leq \|X\| \|X^{-1}\| \max_i \|\text{diag}(Y_{i1}, \dots, Y_{id})\| \quad (53)$$

$$\leq k(X) \sqrt{d} \max_{i,j} Y_{ij} \quad (54)$$

$$= k(X) \sqrt{d} \lambda_{\max}. \quad (55)$$

where  $X$  is the ground-truth matrix defined in (3) and  $k(X) = \frac{\sigma_{\max}(X)}{\sigma_{\min}(X)}$  is the condition number of  $X$ .

Finally, the statement (10) follows from (31), (43),  $\|\hat{x}\| = \|x\|$ , (44), (49), (55) and

$$\|\Delta M\|^2 = \left\| \sum_i \theta_i \Delta M_i \right\|^2 \quad (56)$$

$$= \sum_{j,k} \left| \sum_i \theta_i [\Delta M_i]_{jk} \right|^2 \quad (57)$$

$$\leq \sum_{j,k} \|\theta\|^2 \sum_i [\Delta M_i]_{jk}^2 \quad (58)$$

$$= \|\theta\|^2 \sum_i \|\Delta M_i\|^2 \quad (59)$$

$$\leq d \|\theta\|^2 E^2 \quad (60)$$

where we have used the Cauchy-Schwarz inequality and the definition of  $E$ . In particular, for all higher orders terms contained in  $\Delta^2$ , one has  $o(\|\Delta^2\|) = o(\|x\|^2) + o(\|\Delta M\|^2) = o(E^2)$ .  $\square$

## References

- Anandkumar, A., Ge, R., Hsu, D., Kakade, S. M., and Telgarsky, M. (2012a). Tensor decompositions for learning latent variable models. *CoRR*, abs/1210.7559.
- Anandkumar, A., Hsu, D., Huang, F., and Kakade, S. M. (2012b). Learning high-dimensional mixtures of graphical models. *arXiv preprint arXiv:1203.0697*.
- Anandkumar, A., Hsu, D., and Kakade, S. M. (2012c). A method of moments for mixture models and hidden Markov models. *CoRR*, abs/1203.0683.
- Arora, S., Ge, R., and Moitra, A. (2012). Learning topic models-going beyond SVD. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pages 1–10. IEEE.
- Balle, B., Hamilton, W., and Pineau, J. (2014). Methods of moments for learning stochastic languages: Unified presentation and empirical comparison. In Jebara, T. and Xing, E. P., editors, *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1386–1394. JMLR Workshop and Conference Proceedings.
- Buesing, L., Sahani, M., and Macke, J. H. (2012). Spectral learning of linear dynamics from generalised-linear observations with application to neural population data. In *Advances in neural information processing systems*, pages 1682–1690.
- Chang, J. T. (1996). Full reconstruction of Markov models on evolutionary trees: identifiability and consistency. *Math Biosci*, 137(1):51–73.
- Corless, R. M., Gianni, P. M., and Trager, B. M. (1997). A reordered Schur factorization method for zero-dimensional polynomial systems with multiple roots. pages 133–140. ACM Press.
- De Lathauwer, L., De Moor, B., and Vandewalle, J. (2004). Computation of the canonical decomposition by means of a simultaneous generalized Schur decomposition. *SIAM Journal on Matrix Analysis and Applications*, 26(2):295–327.
- Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1 – 38.
- Favoreel, W., De Moor, B., and Van Overschee, P. (2000). Subspace state space system identification for industrial processes. *Journal of Process Control*, 10(2):149–155.
- Hsu, D. and Kakade, S. M. (2012). Learning gaussian mixture models: Moment methods and spectral decompositions. *CoRR*, abs/1206.5766.
- Hsu, D., Kakade, S. M., and Zhang, T. (2012). A spectral algorithm for learning hidden Markov models. *Journal of Computer and System Sciences*, 78(5):1460 – 1480.
- Konstantinov, M. M., Petkov, P. H., and Christov, N. D. (1994). Nonlocal perturbation analysis of the Schur system of a matrix. *SIAM Journal on Matrix Analysis and Applications*, 15(2):383–392.
- Kuleshov, V., Chaganty, A., and Liang, P. (2015). Tensor factorization via matrix factorization. In *18th International Conference on Artificial Intelligence and Statistics (AISTATS)*.
- Lim, L. H. and Comon, P. (2009). Nonnegative approximations of nonnegative tensors. *Journal of Chemometrics*, 23(7-8):432–441.
- Lindsay, B. G. (1995). Mixture models: theory, geometry and applications. In *NSF-CBMS regional conference series in probability and statistics*, pages 1–163. JSTOR.
- Mossel, E. and Roch, S. (2006). Learning nonsingular phylogenies and hidden Markov models. *The Annals of Applied Probability*, 16(2):583–614.
- Song, J. and Chen, K. C. (2014). Spectacle: Faster and more accurate chromatin state annotation using spectral learning. *bioRxiv*.
- Titterton, D. M. (1985). *Statistical analysis of finite mixture distributions*. Wiley series in probability and mathematical statistics. Wiley, Chichester ; New York.
- Zhang, Y., Chen, X., Zhou, D., and Jordan, M. I. (2014). Spectral methods meet EM: A provably optimal algorithm for crowdsourcing. In *Advances in Neural Information Processing Systems*, pages 1260–1268.
- Zou, J. Y., Hsu, D., Parkes, D. C., and Adams, R. P. (2013). Contrastive learning using spectral methods. In *Advances in Neural Information Processing Systems*, pages 2238–2246.

---

# Semi-described and semi-supervised learning with Gaussian processes

---

**Andreas Damianou**

Dept. of Computer Science & SITraN  
The University of Sheffield  
Sheffield, UK

**Neil D. Lawrence**

Dept. of Computer Science & SITraN  
The University of Sheffield  
Sheffield, UK

## Abstract

Propagating input uncertainty through non-linear Gaussian process (GP) mappings is intractable. This hinders the task of training GPs using uncertain and partially observed inputs. In this paper we refer to this task as “semi-described learning”. We then introduce a GP framework that solves both, the semi-described and the semi-supervised learning problems (where missing values occur in the *outputs*). Auto-regressive state space simulation is also recognised as a special case of semi-described learning. To achieve our goal we develop variational methods for handling semi-described inputs in GPs, and couple them with algorithms that allow for imputing the missing values while treating the uncertainty in a principled, Bayesian manner. Extensive experiments on simulated and real-world data study the problems of iterative forecasting and regression/classification with missing values. The results suggest that the principled propagation of uncertainty stemming from our framework can significantly improve performance in these tasks.

## 1 INTRODUCTION

In many real-world applications missing values can occur in the data, for example when measurements come from unreliable sensors. Correctly accounting for the partially observed instances is important in order to exploit all available information and increase the strength of the inference model. The focus of this paper is on Gaussian process (GP) models that allow for Bayesian, non-parametric inference.

When the missing values occur in the outputs, the corresponding learning task is known as *semi-supervised learning*. For example, consider the task of learning to classify images where the labelled set is much smaller than the total set. Bootstrapping is a potential solution to this problem [Rosenberg et al., 2005], according to which a model

trained on fully observed data imputes the missing outputs. Previous work in semi-supervised GP learning involved the cluster assumption [Lawrence and Jordan, 2005] for classification. Here we consider an approach which uses the manifold assumption [Chapelle et al., 2006; Kingma et al., 2014] which assumes that the observed, complex data are really generated by a compressed, less-noisy latent space.

The other often encountered missing data problem has to do with unobserved *input* features (e.g. missing pixels in input images). In statistics, a popular approach is to impute missing inputs using a combination of different educated guesses [Rubin, 2004]. In machine learning, Ghahramani and Jordan [1994] learn the joint density of the input and output data and integrate over the missing values. For Gaussian process models the missing input case has received only little attention, due to the challenge of propagating the input uncertainty through the non-linear GP mapping. In this paper we introduce the notion of *semi-described learning* to generalise this scenario. Specifically, we define semi-described learning to be the task of learning from inputs that can have missing or uncertain values. Our approach to dealing with missing inputs in semi-described GP learning is, algorithmically, closer to data imputation methods. However, in contrast to past approaches, the missing values are imputed in a fully probabilistic manner by considering explicit distributions in the input space.

Our aim in this paper is to develop a general framework that solves the semi-supervised and semi-described GP learning. We also consider the related *forecasting regression* problem, which is seen as a pipeline where predictions are obtained iteratively in an auto-regressive manner, while propagating the uncertainty across the predictive sequence, as in [Girard et al., 2003; Quiñero-Candela et al., 2003]. Here, we cast the auto-regressive GP learning as a particular type of semi-described learning. We seek to solve all tasks within a single coherent framework that preserves the fully Bayesian property of the GP methodology.

To achieve our goals we need three methodological tools. Firstly, we need approximations allowing us to consider and communicate uncertainty between the inputs and

the outputs of the non-linear GP model. For this, we build on the variational approach of Titsias and Lawrence [2010] which allows for approximately propagating densities throughout the nodes of GP-based directed graphical models. The resulting representation is particularly advantageous, because the whole input domain is now coherently associated with posterior distributions. We can then sample from the input space in a principled manner so as to populate small initial labelled sets in semi-supervised learning scenarios. In that way, we avoid heuristic self-training methods [Rosenberg et al., 2005] that rely on bootstrapping and present problems due to over-confidence. Previously suggested approaches for modelling input uncertainty in GPs also lack the feature of considering an explicit input distribution for both training and test instances. Specifically, [Girard et al., 2003; Quiñonero-Candela et al., 2003] consider the case of input uncertainty *only at test time*. Propagating the test input uncertainty through a non-linear GP results in a non-Gaussian predictive density, but Girard et al. [2003]; Quiñonero-Candela et al. [2003]; Quiñonero-Candela [2004] rely on moment matching to obtain the predictive mean and covariance. On the other hand, Oakley and O’Hagan [2002] do not derive analytic expressions but, rather, develop a scheme based on simulations. McHutchon and Rasmussen [2011] rely on local approximations inside the latent mapping function, rather than modelling the approximate posterior densities directly. Dallaire et al. [2009] do not propagate the uncertainty of the inputs all the way through the GP mapping but, rather, amend the kernel computations to account for the input uncertainty. [Quiñonero-Candela and Roweis, 2003] can be seen as a special case of our developed framework, when the data imputation is performed using a standard GP-LVM [Lawrence, 2006]. Another advantage of our framework is that it allows us to consider different levels of input uncertainty per point and per dimension without, in principle, increasing the danger of under/overfitting, since input uncertainty is modelled through a set of *variational* rather than model parameters.

The second methodological tool needed to achieve our goals has to do with the need to incorporate partial or uncertain observations into the variational framework. For this, we develop a *variational constraint* mechanism which constrains the distribution of the input space given the observed noisy values. This approach is fast, and the whole framework can be incorporated into a parallel inference algorithm [Gal et al., 2014; Dai et al., 2014]. In contrast, Damianou et al. [2011] consider a separate process for modelling the input distribution. However, that approach cannot easily be extended for the data imputation purposes that concern us, since we cannot consider different uncertainty levels per input and per dimension and, additionally, computation scales cubically with the number of datapoints, even within sparse GP frameworks. The constraints framework that we propose is interesting not only as an inference

tool but also as a modelling approach: if the inputs are constrained with the outputs, then we obtain the Bayesian version of the back-constraints framework of Lawrence and Quiñonero Candela [2006] and Ek et al. [2008]. However, in contrast to these approaches, the constraint defined here is a *variational* one, and operates upon a distribution, rather than single points. Zhu et al. [2012] also follow the idea of constraining the posterior distribution with rich side information, albeit for a completely different application. In contrast, Osborne and Roberts [2007] handle partially missing sensor inputs by modelling correlations in the input space through special covariance functions.

Thirdly, the variational methods developed here need to be encapsulated into algorithms that perform data imputation while correctly accounting for the introduced uncertainty. We develop such algorithms after showing how the considered applications can be cast as learning pipelines that rely on correct propagation of uncertainty between each stage.

In summary, our contributions in this paper are the following; firstly, by building on the Bayesian GP-LVM [Titsias and Lawrence, 2010] and developing a variational constraint mechanism we demonstrate how uncertain GP inputs can be explicitly represented as distributions in both training and test time. Secondly, we couple our variational methodology with algorithms that allow us to solve problems associated with partial or uncertain observations: semi-supervised learning, auto-regressive iterative forecasting and, finally, a newly studied type of GP learning which we refer to as “semi-described” learning. We solve these applications within a single framework, allowing for handling the uncertainty in semi-supervised and semi-described problems in a coherent way. The software accompanying this paper can be found at: <http://git.io/A3TN>. This paper extends our previous workshop paper [Damianou and Lawrence, 2014].

## 2 UNCERTAIN INPUTS REFORMULATION OF GP MODELS

Assume a dataset of input–output pairs stored by rows in matrices  $\mathbf{X} \in \mathbb{R}^{n \times q}$  and  $\mathbf{Y} \in \mathbb{R}^{n \times p}$  respectively. Throughout this paper we will denote rows of the above matrices as  $\{\mathbf{y}_{i,:}, \mathbf{x}_{i,:}\}$  and columns (dimensions) as  $\{\mathbf{y}_j, \mathbf{x}_j\}$ , while single elements (e.g.  $y_{i,j}$ ) will be denoted with a double subscript. We first outline the standard GP formulation, where all variables are fully observed. By assuming that outputs are corrupted by zero-mean Gaussian noise, denoted by  $\epsilon_f$ , we obtain the following generative model:

$$y_{i,j} = \mathbf{f}_j(\mathbf{x}_{i,:}) + (\epsilon_f)_{i,j}, \quad (\epsilon_f)_{i,j} \sim \mathcal{N}(0, \beta^{-1}). \quad (1)$$

We place GP priors on the mapping  $\mathbf{f}$ , so that the function instantiations  $\mathbf{F} = \{\mathbf{f}_j\}_{j=1}^p$  follow a Gaussian distribution  $p(\mathbf{f}_j|\mathbf{X}) = \mathcal{N}(\mathbf{f}_j|\mathbf{0}, \mathbf{K})$ , where  $\mathbf{K}$  is the covariance matrix obtained by evaluating the GP covariance function  $k_f$  on

the inputs  $\mathbf{X}$ . Therefore, the model likelihood  $p(\mathbf{Y}|\mathbf{X})$  is:

$$\int_{\mathbf{F}} p(\mathbf{Y}|\mathbf{F})p(\mathbf{F}|\mathbf{X}) = \prod_{j=1}^p \mathcal{N}(\mathbf{y}_j|\mathbf{0}, \mathbf{K} + \beta^{-1}\mathbf{I}). \quad (2)$$

In the other end of the spectrum is the GP-LVM [Lawrence, 2006], where the inputs are fully unobserved (i.e. *latent*). This corresponds to the unsupervised GP setting. In the absence of observed inputs, the likelihood  $p(\mathbf{Y}|\mathbf{X})$  takes the same form as in equation (2) but the inputs  $\mathbf{X}$  now need to be recovered from the outputs  $\mathbf{Y}$  through maximum likelihood. The Bayesian GP-LVM proceeds by additionally placing a Gaussian prior on the latent space,  $p(\mathbf{X}) = \prod_{i=1}^n \mathcal{N}(\mathbf{x}_{i,:}|\mathbf{0}, \mathbf{I})$ , and approximately integrating it out by constructing a variational lower bound  $\mathcal{F}$ , where

$$\mathcal{F} \leq \log p(\mathbf{Y}) = \log \int_{\mathbf{X}} p(\mathbf{Y}|\mathbf{X})p(\mathbf{X}),$$

and by introducing a variational distribution

$$q(\mathbf{X}) = \prod_{i=1}^n q(\mathbf{x}_{i,:}) = \prod_{i=1}^n \mathcal{N}(\mathbf{x}_{i,:}|\boldsymbol{\mu}_{i,:}, \mathbf{S}_{i,:}),$$

where  $\mathbf{S}_{i,:}$  is a diagonal matrix, so that  $\boldsymbol{\mu}_{i,:}, \text{diag}(\mathbf{S}_{i,:}) \in \mathbb{R}^q$ . We can derive an expression for this variational bound,

$$\mathcal{F} = \langle \log p(\mathbf{Y}|\mathbf{X}) \rangle_{q(\mathbf{X})} - \text{KL}(q(\mathbf{X}) \| p(\mathbf{X})), \quad (3)$$

where  $\langle \cdot \rangle_{q(\mathbf{X})}$  denotes an expectation with respect to  $q(\mathbf{X})$ . Since  $\mathbf{X}$  appears non-linearly inside  $p(\mathbf{Y}|\mathbf{X})$  (in the inverse of the covariance matrix  $\mathbf{K} + \beta^{-1}\mathbf{I}$ ), the first term of the above variational bound is intractable. However, we can follow [Titsias and Lawrence, 2010] to approximate the intractable expectation analytically.

In this paper we wish to define a general framework that operates in the whole range of the two aforementioned extrema, i.e. the fully observed and fully unobserved inputs case. The first step to obtaining such a framework is to allow for uncertainty in the inputs. We assume that the inputs  $\mathbf{X}$  are not observed directly but, rather, we only have access to their noisy versions  $\{\mathbf{z}_{i,:}\}_{i=1}^n = \mathbf{Z} \in \mathbb{R}^{n \times q}$ . The relationship between the noisy and true inputs is given by assuming Gaussian noise:

$$\mathbf{x}_{i,:} = \mathbf{z}_{i,:} + (\boldsymbol{\epsilon}_x)_{i,:}, \quad (\boldsymbol{\epsilon}_x)_{i,:} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_x), \quad (4)$$

so that  $p(\mathbf{X}|\mathbf{Z}) = \prod_{i=1}^n \mathcal{N}(\mathbf{x}_{i,:}|\mathbf{z}_{i,:}, \boldsymbol{\Sigma}_x)$ . Obviously, when this distribution collapses to a delta function we recover the standard GP case, and when  $\mathbf{Z}$  is not given we recover the GP-LVM. The problem with the modelling assumption of equation (4) is that now we cannot use equation (1), since the inputs are not available. On the other hand, if we replace  $\mathbf{x}_{i,:}$  in that equation with  $\mathbf{z}_{i,:}$ , then we effectively ignore the input noise. McHutchon and Rasmussen [2011] proceed by combining equations (1) and (4) to obtain the GP mapping  $\mathbf{f}_j(\mathbf{x}_{i,:} - (\boldsymbol{\epsilon}_x)_{i,:})$  which is then

treated using local approximations. However, our aim in this paper is to consider an explicit input distribution. One way to achieve this is to treat the unobserved true inputs as latent variables to be estimated from the marginal likelihood  $p(\mathbf{Y}|\mathbf{Z}) = \int_{\mathbf{X}} p(\mathbf{Y}|\mathbf{X})p(\mathbf{X}|\mathbf{Z})$ . Following Damianou et al. [2011] we can obtain a variational lower bound  $\mathcal{F} \leq \log p(\mathbf{Y}|\mathbf{Z})$ , with:

$$\mathcal{F} = \langle \log p(\mathbf{Y}|\mathbf{X}) \rangle_{q(\mathbf{X})} - \text{KL}(q(\mathbf{X}) \| p(\mathbf{X}|\mathbf{Z})). \quad (5)$$

This formulation corresponds to the graphical model of Figure 1(a). However, with this approach one needs to additionally estimate the noise parameters  $\boldsymbol{\Sigma}_x$ , which might be challenging given their large number and their interplay with the variational noise parameters  $\{\mathbf{S}_{i,:}\}_{i=1}^n$ . Therefore we considered an alternative solution which we found to result in better performance.

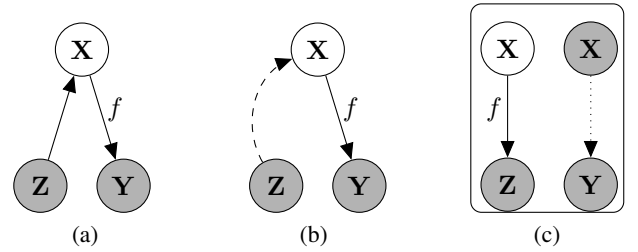


Figure 1: Incorporating uncertain inputs  $\mathbf{Z}$  in GPs through an intermediate input space  $\mathbf{X}$  by considering: (a) a Gaussian prior on  $\mathbf{X}$ , centered on  $\mathbf{Z}$  and (b) a variational constraint (dashed line) on the approximate posterior. Figure (c) represents our two-stage approach to dealing with missing outputs for classification, where the dotted line represents a discriminative mapping.

## 2.1 VARIATIONAL CONSTRAINT

An alternative way of relating the true with the noisy inputs can be obtained by focusing on the *posterior* rather than the prior distribution. To start with, we re-express the variational lower bound of equation (5) as:

$$\log p(\mathbf{Y}|\mathbf{Z}) \geq \int_{\mathbf{X}} q(\mathbf{X}) \log \frac{p(\mathbf{Y}|\mathbf{Z})p(\mathbf{X}|\mathbf{Y}, \mathbf{Z})}{q(\mathbf{X})} = \mathcal{F}$$

from where we break the logarithm to obtain:

$$\mathcal{F} = \log p(\mathbf{Y}|\mathbf{Z}) - \text{KL}(q(\mathbf{X}) \| p(\mathbf{X}|\mathbf{Y}, \mathbf{Z})).$$

We see that the lower bound becomes exact when the variational distribution  $q(\mathbf{X})$  matches the true posterior distribution of the noise-free latent inputs given the observed inputs and outputs. To allow for this approximation we introduce a simple variational constraint which operates on the factorised distribution, which is now written as  $q(\mathbf{X}|\mathbf{Z})$  to highlight its dependency on  $\mathbf{Z}$ . In the simplest case



where all inputs are observed but uncertain, the constraint just consists of replacing the variational means  $\mu_{i,:}$  of each factor  $q(\mathbf{x}_{i,:})$  with the corresponding observed input  $\mathbf{z}_{i,:}$ . The variational parameters  $\mathbf{S}_{i,:}$  then account for the uncertainty. Similarly to the back-constraint of Lawrence and Quiñonero Candela [2006]; Ek et al. [2008], our variational constraint does not constitute a probabilistic mapping. However, it allows us to encode the input noise directly in the approximate posterior without having to specify additional noise parameters or sacrifice scalability. Next, we elaborate on the exact form of the constraint.

In the general case, namely having inputs that are only partially observed, we can define a similar constraint which specifies a variational distribution as a mix of Gaussian and Dirac delta distributions. Notationally we consider data to be split into fully and partially observed subsets, e.g.  $\mathbf{Z} = (\mathbf{Z}^\circ, \mathbf{Z}^\mathcal{U})$ , where  $\circ$  and  $\mathcal{U}$  denote fully and partially observed sets respectively. The features missing in  $\mathbf{Z}^\mathcal{U}$  can appear in different dimension(s) for each individual point  $\mathbf{z}_{i,:}^\mathcal{U}$ , but for notational clarity  $\mathcal{U}$  will index rows containing at least one missing dimension. In this case, the variational distribution is constrained to have the form

$$q(\mathbf{X}|\mathbf{Z}, \{\circ, \mathcal{U}\}) = q(\mathbf{X}^\circ|\mathbf{Z}^\circ) q(\mathbf{X}^\mathcal{U}|\mathbf{Z}^\mathcal{U}) \\ = \prod_{i \in \circ} \mathcal{N}(\mathbf{x}_{i,:}^\circ | \mathbf{z}_{i,:}^\circ, \varepsilon \mathbf{I}) \prod_{i \in \mathcal{U}} \mathcal{N}(\mathbf{x}_{i,:}^\mathcal{U} | \boldsymbol{\mu}_{i,:}^\mathcal{U}, \mathbf{S}_{i,:}^\mathcal{U}), \quad (6)$$

where  $\varepsilon \rightarrow 0$ , so that the corresponding distributions approximate a Dirac delta. Notice that for a partially observed row  $\mathbf{z}_{i,:}^\mathcal{U}$ , we can still replace an observed dimension  $j$  with its corresponding observation in the second set of factors of equation (6), i.e.  $\mu_{i,j}^\mathcal{U} = z_{i,j}^\mathcal{U}$ , so  $q(\mathbf{X}^\mathcal{U}|\mathbf{Z}^\mathcal{U}) \neq q(\mathbf{X}^\mathcal{U})$ . Given the above, as well as a spherical Gaussian prior for  $p(\mathbf{X})$ , the required intractable density  $\log p(\mathbf{Y}|\mathbf{Z})$  is approximated with a variational lower bound:

$$\mathcal{F} = \langle \log p(\mathbf{Y}|\mathbf{X}) \rangle_{q(\mathbf{X}|\mathbf{Z})} - \text{KL}(q(\mathbf{X}|\mathbf{Z}) \| p(\mathbf{X})), \quad (7)$$

where for clarity we dropped the dependency on  $\{\circ, \mathcal{U}\}$  from our expressions. Since the Dirac functions are approximated with sharply peaked Gaussians inside the posterior  $q(\mathbf{X}|\mathbf{Z})$ , the above variational bound can be computed in the same manner as the Bayesian GP-LVM bound of equation (3). Specifically, the KL term is tractable, since it only involves Gaussians.

As for the first term of equation (7), we follow the Bayesian GP-LVM methodology and we augment the probability space with  $m$  extra samples  $\mathbf{U} = \{\mathbf{u}_i\}_{i=1}^m$  of the latent function  $f$  evaluated at a set of pseudo-inputs (known as ‘‘inducing points’’)  $\mathbf{X}_u$ , so that  $\mathbf{U} \in \mathbb{R}^{m \times p}$  and  $\mathbf{X}_u \in \mathbb{R}^{m \times q}$ . Due to the consistency of GPs,  $p(\mathbf{U}|\mathbf{X}_u)$  is a Gaussian distribution. From now on we omit dependence on  $\mathbf{X}_u$  from our expressions. The likelihood then becomes:

$$p(\mathbf{Y}, \mathbf{F}, \mathbf{U}|\mathbf{X}) = p(\mathbf{Y}|\mathbf{F})p(\mathbf{F}|\mathbf{U}, \mathbf{X})p(\mathbf{U}).$$

Then, the marginal  $p(\mathbf{Y}|\mathbf{X})$  can be obtained from Jensen’s inequality after introducing a variational distribution  $q(\mathbf{F}, \mathbf{U})$ , so that  $\hat{\mathcal{F}} \leq \log p(\mathbf{Y}|\mathbf{X})$ , where:

$$\hat{\mathcal{F}} = \int_{\mathbf{F}, \mathbf{U}} q(\mathbf{F}, \mathbf{U}) \log \frac{p(\mathbf{Y}|\mathbf{F})p(\mathbf{F}|\mathbf{U}, \mathbf{X})p(\mathbf{U})}{q(\mathbf{F}, \mathbf{U})}. \quad (8)$$

Now the first term of equation (7) is approximated as  $\langle p(\mathbf{Y}|\mathbf{X}) \rangle_{q(\mathbf{X}|\mathbf{Z})} \geq \langle \hat{\mathcal{F}} \rangle_{q(\mathbf{X}|\mathbf{Z})}$ . However, this approximation is still intractable, since the problematic term  $p(\mathbf{F}|\mathbf{U}, \mathbf{X})$  still appears inside  $\hat{\mathcal{F}}$  and contains  $\mathbf{X}$  in the inverse of the covariance matrix, thus rendering the expectation intractable. The trick of Titsias and Lawrence [2010] is to define a variational distribution of the form:

$$q(\mathbf{F}, \mathbf{U}) = p(\mathbf{F}|\mathbf{U}, \mathbf{X})q(\mathbf{U}). \quad (9)$$

Replacing equation (9) inside the bound of equation (8) results in the cancellation of  $p(\mathbf{F}|\mathbf{U}, \mathbf{X})$ , leaving us with a tractable (partial) bound, which takes the form:

$$\langle p(\mathbf{Y}|\mathbf{X}) \rangle_{q(\mathbf{X}|\mathbf{Z})} \geq \langle \hat{\mathcal{F}} \rangle_{q(\mathbf{X}|\mathbf{Z})} = -\text{KL}(q(\mathbf{U}) \| p(\mathbf{U})) \\ + \int_{\mathbf{X}, \mathbf{U}} \left[ q(\mathbf{X}|\mathbf{Z})q(\mathbf{U}) \int_{\mathbf{F}} p(\mathbf{F}|\mathbf{U}, \mathbf{X}) \log p(\mathbf{Y}|\mathbf{F}) \right] \quad (10)$$

The augmentation trick decouples the latent function values given the inducing points, so that any uncertainty in the inputs can be propagated through the nested integral. After this operation, the inducing outputs  $\mathbf{U}$  can be marginalised out. Therefore, the above integral is analytically tractable, since the nested integral is tractable and results in a Gaussian where  $\mathbf{X}$  no longer appears in the inverse of the covariance matrix. The final lower bound to use as an objective function is thus obtained by using the partial bound of eq. (10) in place of the first term of equation (7), thus obtaining a new, final bound (more details in the Appendix):

$$\mathcal{F}_2 = \langle \hat{\mathcal{F}} \rangle_{q(\mathbf{X}|\mathbf{Z})} - \text{KL}(q(\mathbf{X}|\mathbf{Z}) \| p(\mathbf{X})). \quad (11)$$

To summarise, the variational methodology seeks to approximate the true posterior with a variational distribution  $q(\mathbf{F}, \mathbf{U}, \mathbf{X}) = q(\mathbf{F})q(\mathbf{U})q(\mathbf{X})$ . To achieve this,  $q(\mathbf{F})$  is constrained to take the exact form  $p(\mathbf{F}|\mathbf{U}, \mathbf{X})$ . This term is then ‘‘eliminated’’, giving us tractability, but its effect is re-introduced through the variational distribution (in the nested integral of eq. (10)). Contrast this with the variational constraint on  $q(\mathbf{X})$ : that approximate posterior factor is constrained according to  $\mathbf{Z}$ , so that the effect of  $\mathbf{Z}$  is considered only through the  $q(\mathbf{X}|\mathbf{Z})$  (eq. (11)). The above comparison gives insight in the conceptual similarity of the variational approach followed to obtain tractability and the one followed for handling partially observed inputs.

The variationally constrained model is shown in fig. 1(b). The total set of parameters to be optimised in the objective function  $\mathcal{F}_2$  of equation (11) (e.g. using a gradient-based optimiser) are the model parameters  $(\boldsymbol{\theta}_f, \beta)$ , where  $\boldsymbol{\theta}_f$  denotes the hyper-parameters of the covariance function  $k_f$ ,

and the variational parameters  $(\mathbf{X}_u, \{\boldsymbol{\mu}_{i,:}^u, \mathbf{S}_{i,:}^u\}_{i \in \mathcal{U}})$  ( $q(\mathbf{U})$  can be optimally eliminated, see Appendix). Depending on the application and corresponding learning algorithm, certain dimensions of  $\{\boldsymbol{\mu}_{i,:}^u, \mathbf{S}_{i,:}^u\}$  can be treated as observed. Such algorithms are discussed in the following sections.

### 3 GP LEARNING WITH MISSING VALUES

We formulate both the semi-described and semi-supervised learning as particular instances of learning a mapping function where the inputs are associated with uncertainty. In both cases, we devise a two-step strategy based on our uncertain inputs GP framework, which allows to efficiently take into account the partial information in the given datasets to improve the predictive performance. For brevity, we refer to the framework described in the previous section as a *variationally constrained GP*, from where a semi-described, an auto-regressive and a semi-supervised GP approach are obtained as special cases, given the algorithms that will be explained in this section.

#### 3.1 SEMI-DESCRIBED LEARNING

We assume a set of observed outputs  $\mathbf{Y}$  that correspond to fully observed inputs  $\mathbf{Z}^\circ$  and partially observed inputs  $\mathbf{Z}^u$ , so that  $\mathbf{Z} = (\mathbf{Z}^\circ, \mathbf{Z}^u)$ . To make the correspondence clearer, we also split the observed *outputs* according to the sets  $\{\circ, u\}$ , so that  $\mathbf{Y} = (\mathbf{Y}^\circ, \mathbf{Y}^u)$ , but note that both output sets are fully observed. We are then interested in learning a regression function from  $\mathbf{Z}$  to  $\mathbf{Y}$  by using all available information. Since in the variationally constrained GP the inputs are replaced by distributions  $q(\mathbf{X}^\circ | \mathbf{Z}^\circ)$  and  $q(\mathbf{X}^u | \mathbf{Z}^u)$ , the uncertainty over  $\mathbf{Z}^u$  can be taken into account naturally through this variational distribution. In this context, we formulate a data imputation-based approach which is inspired by self-training methods; nevertheless, it is more principled in the handling of uncertainty.

Specifically, the algorithm has two stages; in the first step, we use the fully observed data subset  $(\mathbf{Z}^\circ, \mathbf{Y}^\circ)$  to train an initial variationally constrained GP model which encapsulates the sharply peaked variational distribution  $q(\mathbf{X}^\circ | \mathbf{Z}^\circ)$  given in equation (6). Given this model, we can then use  $\mathbf{Y}^u$  to estimate the predictive posterior<sup>1</sup>  $q(\mathbf{X}^u | \mathbf{Z}^u)$  in the missing locations of  $\mathbf{Z}^u$  (for the observed locations we match the mean with the observations in a sharply peaked marginal, as for  $\mathbf{Z}^\circ$ ). Essentially, we replace the missing locations of the variational means  $\boldsymbol{\mu}_{i,:}^u$  and variances  $\mathbf{S}_{i,:}^u$  of  $q(\mathbf{X}^u | \mathbf{Z}^u)$  with the predictive mean and variance obtained through the “self-training” step. This selection for  $\{\boldsymbol{\mu}_{i,:}^u, \mathbf{S}_{i,:}^u\}$  constitutes nevertheless only an initialisation. In

<sup>1</sup>The predictive posterior for test data  $\mathbf{Y}_*$  is obtained by maximising a variational lower bound similar to the training one (eq. (11)), but  $\mathbf{X}$  and  $\mathbf{Y}$  are now replaced with  $(\mathbf{X}, \mathbf{X}_*)$  and  $(\mathbf{Y}, \mathbf{Y}_*)$ .

the next step, these parameters are further optimised together with the fully observed data. Specifically, after initializing  $q(\mathbf{X} | \mathbf{Z}) = q(\mathbf{X}^\circ, \mathbf{X}^u | \mathbf{Z})$  as explained in step 1, we proceed to train a variationally constrained GP model on the full (extended) training set  $((\mathbf{Z}^\circ, \mathbf{Z}^u), (\mathbf{Y}^\circ, \mathbf{Y}^u))$ , which contains fully and partially observed inputs.

Algorithm 1 outlines the approach in more detail. This formulation defines a *semi-described GP* approach which naturally incorporates fully and partially observed examples by communicating the uncertainty throughout the relevant parts of the model in a principled way. Indeed, the predictive uncertainty obtained when imputing missing values in the first step of the pipeline is incorporated as input uncertainty in the second step of the pipeline. In extreme cases resulting in very non-confident predictions, for example in the presence of outliers, the corresponding locations will simply be ignored automatically due to the large uncertainty. This mechanism, together with the subsequent optimisation of the parameters of  $q(\mathbf{X}^u | \mathbf{Z}^u)$  in stage 2, guards against reinforcing bad predictions when imputing missing values based on a smaller training set. The model includes GP regression and the GP-LVM as special cases. In particular, in the limit of having no observed values our semi-described GP is equivalent to the GP-LVM and when there are no missing values it is equivalent to GP regression.

There are some similarities to traditional self-training [Rosenberg et al., 2005], but as there are no straightforward mechanisms to propagate uncertainty in that domain, they typically rely on bootstrapping followed by thresholding “bad” samples to prevent model over-confidence. In our framework, the predictions made by the initial model only constitute initialisations which are later optimised along with model parameters and, hence, we refer to this step as *partial* self-training. Further, the predictive uncertainty is not used as a hard measure of discarding unconfident predictions; instead, we allow all values to contribute according to an optimised uncertainty measure, that is, the input variances  $\mathbf{S}_i$ . Therefore, the way in which uncertainty is handled makes the self-training part of our algorithm principled compared to many bootstrap-based approaches.

#### DEMONSTRATION

We considered simulated and real-world data to demonstrate our semi-described GP algorithm. The simulated data were created by sampling inputs  $\mathbf{Z}$  from a GP (which was unknown to the competing models) and then giving these samples as input to another unknown GP, to obtain corresponding outputs  $\mathbf{Y}$ . For the real-world data demonstration we considered a motion capture dataset taken from subject 35 in the CMU motion capture database. We selected a subset of walk and run motions of a human body represented as a set of 59 joint locations. We formulated a regression problem where the first 20 dimensions of the original data are used as targets and the remaining 39 as

---

**Algorithm 1** Semi-described learning with uncertain input GPs.

---

- 1: *Given:* Fully and partially observed inputs,  $\mathbf{Z}^\circ$  and  $\mathbf{Z}^\mathcal{U}$  respectively, corresponding to fully observed outputs  $\mathbf{Y}^\circ$  and  $\mathbf{Y}^\mathcal{U}$ .
  - 2: Construct  $q(\mathbf{X}^\circ|\mathbf{Z}^\circ) = \prod_{i=1}^n \mathcal{N}(\mathbf{x}_{i,:}^\circ|\mathbf{z}_{i,:}^\circ, \varepsilon\mathbf{I})$ , where:  $\varepsilon \rightarrow 0$
  - 3: Fix  $q(\mathbf{X}^\circ|\mathbf{Z}^\circ)$  in the optimiser # (i.e.  $q(\mathbf{X}^\circ|\mathbf{Z}^\circ)$  has no free parameters)
  - 4: Train a variationally constrained GP model  $\mathcal{M}^\circ$  with inputs  $q(\mathbf{X}^\circ|\mathbf{Z}^\circ)$  and outputs  $\mathbf{Y}^\circ$
  - 5: **for**  $i = 1, \dots, |\mathbf{Y}^\mathcal{U}|$  **do**
  - 6:   Predict the distribution  $\mathcal{N}(\mathbf{x}_{i,:}^\mathcal{U}|\hat{\boldsymbol{\mu}}_{i,:}^\mathcal{U}, \hat{\mathbf{S}}_i^\mathcal{U}) \approx p(\mathbf{x}_{i,:}^\mathcal{U}|\mathbf{y}_{i,:}^\mathcal{U}, \mathcal{M}^\circ)$  from the approximate posterior of model  $\mathcal{M}^\circ$ .
  - 7:   Initialise parameters  $\{\boldsymbol{\mu}_{i,:}^\mathcal{U}, \mathbf{S}_i^\mathcal{U}\}$  of  $q(\mathbf{x}_{i,:}^\mathcal{U}|\mathbf{z}_{i,:}^\mathcal{U}) = \mathcal{N}(\mathbf{x}_{i,:}^\mathcal{U}|\boldsymbol{\mu}_{i,:}^\mathcal{U}, \mathbf{S}_i^\mathcal{U})$  as follows:
  - 8:   **for**  $j = 1, \dots, q$  **do**
  - 9:     **if**  $z_{i,j}^\mathcal{U}$  is observed **then**
  - 10:        $\mu_{i,j}^\mathcal{U} = z_{i,j}^\mathcal{U}$  and  $(\mathbf{S}_i^\mathcal{U})_{j,j} = \varepsilon$ , where:  $\varepsilon \rightarrow 0$
  - 11:       Fix  $\mu_{i,j}^\mathcal{U}, (\mathbf{S}_i^\mathcal{U})_{j,j}$  in the optimiser # (i.e. they don't constitute parameters)
  - 12:     **else**
  - 13:        $\mu_{i,j}^\mathcal{U} = \hat{\mu}_{i,j}^\mathcal{U}$  and  $(\mathbf{S}_i^\mathcal{U})_{j,j} = (\hat{\mathbf{S}}_i^\mathcal{U})_{j,j}$
  - 14: Train model  $\mathcal{M}^{\circ,\mathcal{U}}$  with inputs  $q(\mathbf{X}^{\{\circ,\mathcal{U}\}}|\mathbf{Z}^{\{\circ,\mathcal{U}\}})$  and outputs  $(\mathbf{Y}^\circ, \mathbf{Y}^\mathcal{U})$ . The input distribution  $q(\mathbf{X}^{\{\circ,\mathcal{U}\}}|\mathbf{Z}^{\{\circ,\mathcal{U}\}}) = q(\mathbf{X}^\circ|\mathbf{Z}^\circ)q(\mathbf{X}^\mathcal{U}|\mathbf{Z}^\mathcal{U})$  is constructed in steps 2, 5-13 and further optimised in the non-fixed locations.
  - 15: Model  $\mathcal{M}^{\circ,\mathcal{U}}$  now constitutes the semi-described GP and can be used for all required prediction tasks.
- 

inputs. That is, given a partial joint representation of the human body, the task is to infer the rest of the representation. For both datasets, simulated and motion capture, we selected a portion of the training inputs, denoted as  $\mathbf{Z}^\mathcal{U}$ , to have randomly missing features. The extended dataset  $((\mathbf{Z}^\circ, \mathbf{Z}^\mathcal{U}), (\mathbf{Y}^\circ, \mathbf{Y}^\mathcal{U}))$  was used to train: a) our method, referred to as semi-described GP (SD-GP) b) multiple linear regression (MLR) c) regression by performing nearest neighbour (NN) search between the test and training instances, in the observed input locations d) performing data imputation using the standard GP-LVM. Not taking into account the predictive uncertainty during imputation was found to have catastrophic results in the simulated data, as the training set was not robust against bad predictions. Therefore, the ‘‘GP-LVM’’ variant was not considered in the real data experiment. We also considered: e) a standard GP which cannot handle missing inputs straightforwardly and so was trained only on the observed data  $(\mathbf{Z}^\circ, \mathbf{Y}^\circ)$ . The goal was to reconstruct test outputs  $\mathbf{Y}_*$  given fully observed test inputs  $\mathbf{Z}_*$ . For the simulated data we used the following sizes:  $|\mathbf{Z}^\circ| = 40$ ,  $|\mathbf{Z}^\mathcal{U}| = 60$  and  $|\mathbf{Z}_*| = 100$ . The dimensionality of the inputs is  $q = 15$  and of the outputs is  $p = 5$ . For the motion capture data we used  $|\mathbf{Z}^\circ| = 50$ ,  $|\mathbf{Z}^\mathcal{U}| = 80$  and  $|\mathbf{Z}_*| = 200$ . In fig. 2 we plot the MSE obtained by the competing methods for a varying percentage of missing features in  $\mathbf{Z}^\mathcal{U}$ . For the simulated data experiment, each of the points in the plot is an average of 4 runs which considered different random seeds. For clarity, the  $y$ -axis limit is fixed in figure 2, because some methods produced huge errors. The full picture is in figure 5 (Appendix). As can be seen in the figures, the semi-described GP is able to handle the extra data and make much better predictions, even if a very large portion is missing. Indeed, its performance starts to converge to that of a standard GP when there are 90% missing values in  $\mathbf{Z}^\mathcal{U}$  and performs

identically to the standard GP when 100% of the values are missing. We found that when  $q$  is large compared to  $p$  and  $n$ , then the data imputation step can be problematic as the percentage of missing features in  $\mathbf{Z}^\mathcal{U}$  approaches 100% i.e. the method is reliant on having some covariates available. Appendix D discusses this behaviour, but a more systematic investigation is left as future work.

### 3.2 AUTO-REGRESSIVE GAUSSIAN PROCESSES

Having a method which implicitly models the uncertainty in the inputs of a GP also allows for doing predictions in an autoregressive manner [Oakley and O’Hagan, 2002] while propagating the uncertainty through the predictive sequence [Girard et al., 2003; Quiñonero-Candela et al., 2003]. Specifically, assuming that the given data  $\mathbf{Y}$  constitute a multivariate timeseries where the observed time vector  $\mathbf{t}$  is equally spaced, and given a time-window of length  $\tau$ , we can reformat  $\mathbf{Y}$  into input-output collections of pairs  $\hat{\mathbf{Z}}$  and  $\hat{\mathbf{Y}}$  as follows: the first input to the model,  $\hat{\mathbf{z}}_{1,:}$ , will be given by the stacked vector  $[\mathbf{y}_{1,:}, \dots, \mathbf{y}_{\tau,:}]$  and the first output,  $\hat{\mathbf{y}}_{1,:}$ , will be given by  $\mathbf{y}_{\tau+1,:}$  and similarly for the other data in  $\hat{\mathbf{Z}}$  and  $\hat{\mathbf{Y}}$ , so that:

$$\begin{aligned} [\hat{\mathbf{z}}_{1,:}, \hat{\mathbf{z}}_{2,:}, \dots, \hat{\mathbf{z}}_{n-\tau,:}] &= \\ & [[\mathbf{y}_{1,:}, \mathbf{y}_{2,:}, \dots, \mathbf{y}_{\tau,:}], [\mathbf{y}_{2,:}, \mathbf{y}_{3,:}, \dots, \mathbf{y}_{\tau+1,:}], \dots], \\ [\hat{\mathbf{y}}_{1,:}, \hat{\mathbf{y}}_{2,:}, \dots, \hat{\mathbf{y}}_{n-\tau,:}] &= [\mathbf{y}_{\tau+1,:}, \mathbf{y}_{\tau+2,:}, \dots, \mathbf{y}_{n,:}]. \end{aligned}$$

To perform extrapolation we first train the model on the modified dataset  $(\hat{\mathbf{Z}}, \hat{\mathbf{Y}})$ . By referring to the semi-described formulation described in Section 3.1, we assign all training inputs to the observed set  $\mathcal{O}$ . After training, we can perform iterative prediction to find a future sequence  $\hat{\mathbf{Z}}_* := [\mathbf{y}_{n+1,:}, \mathbf{y}_{n+2,:}, \dots]$  where, similarly to the approach taken by Girard et al. [2003], the predictive variance in each

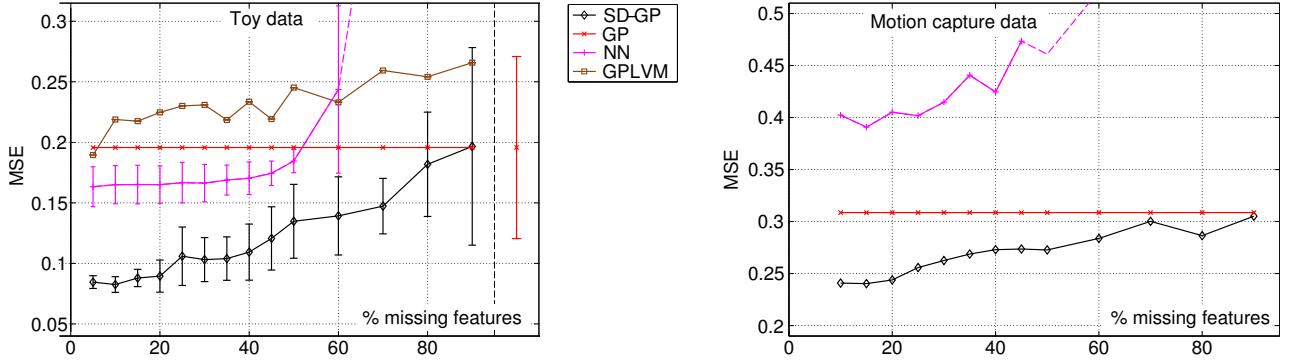


Figure 2: MSE for predictions obtained by different methods on semi-described learning. GP cannot handle partial observations, thus the uncertainty ( $2\sigma$ ) is constant; for clarity, the errorbar is plotted separately on the right of the dashed vertical line (for nonsensical  $x$  values). The results for simulated data are obtained from 4 trials. For clarity, the limits on the  $y$ -axis are fixed, so when the errors become too big for certain methods they get off the chart. The errorbars for the GPLVM-based approach are also too large and not plotted. The full picture is given in figure 5 (Appendix).

step is accounted for and propagated in the subsequent predictions. The algorithm makes iterative 1-step predictions in the future; initially, the output  $\hat{\mathbf{z}}_{1,*} := \mathbf{y}_{n+1,:}$  will be predicted (given the training set) with predictive variance  $\hat{\mathbf{S}}_{*,1}$ . In the next step, the “observations” set will be augmented to include the distribution of predictions over  $\mathbf{y}_{n+1,:}$ , by defining  $q(\mathbf{x}_{n+1,:} | \hat{\mathbf{z}}_{1,*}) = \mathcal{N}(\mathbf{x}_{n+1,:} | \hat{\mathbf{z}}_{*,1}, \hat{\mathbf{S}}_{*,1})$ , and so on. This simulation process can be seen as constructing a predictive sequence step by step, i.e. the newly inserted input points constitute parts of the (test) predictive sequence and not training points. Therefore, this procedure can be seen as an iterative version of semi-described learning.

Note that it is straightforward to extend this model by applying this auto-regressive mechanism in a latent space of a stacked model or, more generally, as a deep GP [Damianou and Lawrence, 2013]. By additionally introducing functions that map from this latent space nonlinearly to an observation space, we obtain a fully nonlinear state space model in the manner of Deisenroth et al. [2012]. For our model, uncertainty is encoded in both the states and the nonlinear transition functions. Correct propagation of uncertainty is vital in well calibrated models of future system behavior, and automatic determination of the structure of the model (e.g. the window size) can be informative in describing the order of the underlying dynamical system.

### DEMONSTRATION: ITERATIVE FORECASTING

Here we demonstrate our framework in the simulation of a state space model. We consider the Mackey-Glass chaotic time series, a standard benchmark which was also considered by Girard et al. [2003]. The data is one-dimensional so that the timeseries can be represented as pairs of values  $\{\mathbf{y}, \mathbf{t}\}$ ,  $t = 1, 2, \dots, n$  and simulates the process:

$$\frac{d\zeta(t)}{dt} = -b\zeta(t) + \alpha \frac{\zeta(t-T)}{1+\zeta(t-T)^{10}}, (\alpha, b, T) = (0.2, 0.1, 17).$$

Obviously the generating process is very non-linear, rendering this dataset challenging. We trained the autoregressive model on data from this series, where the modified dataset  $\{\hat{\mathbf{z}}, \hat{\mathbf{y}}\}$  was created with  $\tau = 18$  and we used the first  $4\tau = 72$  points to train the model and predicted the subsequent 1110 points through iterative free simulation.

We compared our method with a “naive autoregressive” GP model where the input-output pairs were given by the autoregressive modification of the dataset  $\{\hat{\mathbf{z}}, \hat{\mathbf{y}}\}$ . For that model, the predictions are made iteratively and the predicted values after each predictive step are added to the “observation” set. However, this standard GP model has no straight forward way of incorporating/propagating the uncertainty and, therefore, the input uncertainty is zero for every step of the iterative predictions. We also compared against the method of Girard et al. [2003]<sup>2</sup>, denoted in the plots as “GP<sub>uncert</sub>”. Figure 3 shows the results for the last 310 steps (i.e.  $t = 800$  onwards) of the full free simulation (1110-step ahead forecasting); figure 6 (Appendix) gives a more complete picture. As can be seen in the variances plot, both our method and GP<sub>uncert</sub> are more robust in handling the uncertainty throughout the predictions; the “naive” GP method underestimates the uncertainty. Consequently, as can be seen in figure 6, in the first few predictions all methods give the same answer. However, once the predictions of the “naive” method diverge a little by the true values, the error is carried on and amplified due to underestimating the uncertainty. On the other hand, GP<sub>uncert</sub> perhaps overestimates the uncertainty and, therefore, is more conservative in its predictions, resulting in higher errors. Quantification of the error is shown in Table 1 (Appendix).

<sup>2</sup>We implemented the basic moment matching approach, although in the original paper the authors use additional approximations, namely Taylor expansion around the predictive moments.

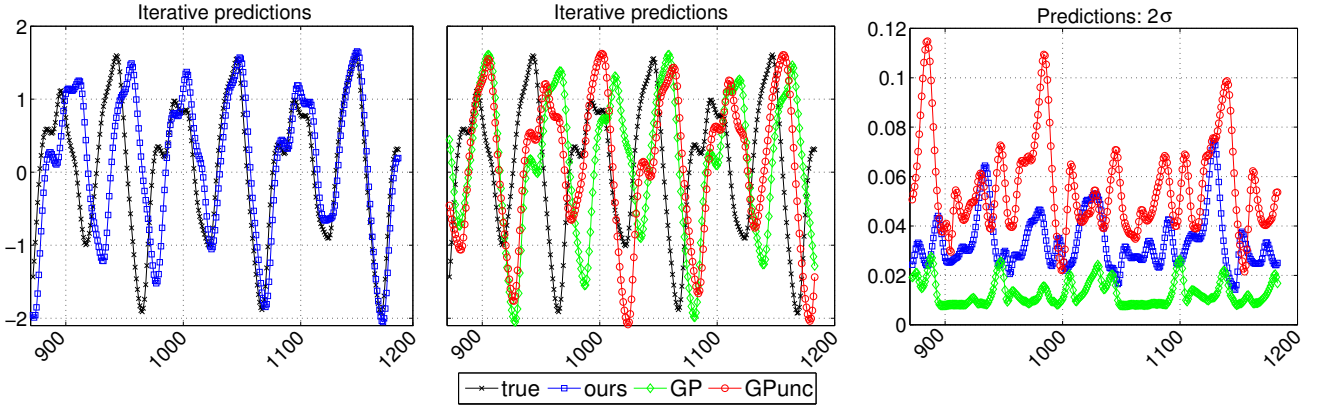


Figure 3: Chaotic timeseries: forecasting 1110 steps ahead by iterative prediction. The first 800 steps are not shown here, but figure 6 (Appendix) gives the complete picture. Comparing: a “naive autoregressive” GP which does not propagate (and hence underestimates) the uncertainties; the method of Girard et al. [2003], referred to as  $\text{GP}_{\text{uncert}}$ ; and our approach, which closely tracks the true test sequence until the last steps of the extrapolation. The comparative depiction of the predictions is split into two plots (for clarity), left and center. The rightmost plot shows the predictive uncertainties ( $2\sigma$ ).  $x$ -axis is the prediction step ( $t$ ) and  $y$ -axis is the function value,  $f(t)$ .

### 3.3 SEMI-SUPERVISED LEARNING

In this section we study semi-supervised learning which, in contrast to semi-described learning, is for handling missing values in the outputs. This scenario is typically encountered in classification settings. Therefore, we introduce the sets  $\{\mathcal{L}, \mathcal{M}\}$  that index respectively the *labelled* and *missing* (unlabelled) rows of the outputs (labels)  $\mathbf{Y}$ . Accordingly, the full dataset is split so that  $\mathbf{Z} = (\mathbf{Z}^{\mathcal{L}}, \mathbf{Z}^{\mathcal{M}})$  and  $\mathbf{Y} = (\mathbf{Y}^{\mathcal{L}}, \mathbf{Y}^{\mathcal{M}})$ , where  $\mathbf{Z}$  is now fully observed. The task is then to devise a method that improves classification performance by using both labelled and unlabelled data.

Inspired by Kingma et al. [2014] we define a semi-supervised GP framework where features are extracted from all available information and, subsequently, are given as inputs to a discriminative classifier. Specifically, using the whole input space  $\mathbf{Z}$ , we learn a low-dimensional latent space  $\mathbf{X}$  through an approximate posterior  $q(\mathbf{X}) \approx p(\mathbf{X}|\mathbf{Z})$ . Obviously, this specific case where the input space is uncertain but totally unobserved (i.e. a latent space) just reduces to the Bayesian GP-LVM model. Notice that the posterior  $q(\mathbf{X})$  is no longer constrained with  $\mathbf{Z}$  but, rather, directly approximates  $p(\mathbf{X}|\mathbf{Z})$ , since we now have a forward *probabilistic* mapping from  $\mathbf{X}$  to  $\mathbf{Z}$  and  $\mathbf{Z}$  is treated as a random variable with  $p(\mathbf{Z}|\mathbf{X})$  being a Gaussian distribution, i.e. exactly the same setting used in the GP-LVM. Since there is one-to-one correspondence between  $\mathbf{X}$ ,  $\mathbf{Z}$  and  $\mathbf{Y}$ , we can notationally write  $\mathbf{X} = (\mathbf{X}^{\mathcal{L}}, \mathbf{X}^{\mathcal{M}})$ . Further, since we assume that  $q(\mathbf{X})$  is factorised across datapoints, we can write  $q(\mathbf{X}) = q(\mathbf{X}^{\mathcal{L}})q(\mathbf{X}^{\mathcal{M}})$ .

In the second step of our semi-supervised algorithm, we train a discriminative classifier from  $q(\mathbf{X}^{\mathcal{L}})$  to the observed labelled space,  $\mathbf{Y}^{\mathcal{L}}$ . The main idea is that, by including the inputs  $\mathbf{Z}^{\mathcal{M}}$  in the first learning step, we manage to de-

fine a better latent embedding from which we can extract a more useful set of features for the discriminative classifier. Notice that what we would ideally use as input to the discriminative classifier is a whole distribution, rather than single point estimates. Therefore, we wish to take advantage of the associated uncertainty; specifically, we can populate the labelled set by sampling from the distribution  $q(\mathbf{X}^{\mathcal{L}})$ . For example, if a latent point  $\mathbf{x}_{i,:}^{\mathcal{L}}$  corresponds to the input-output pair  $(\mathbf{z}_{i,:}^{\mathcal{L}}, \mathbf{y}_{i,:}^{\mathcal{L}})$ , then a sample from  $q(\mathbf{x}_{i,:}^{\mathcal{L}})$  will be assigned the label  $\mathbf{y}_{i,:}^{\mathcal{L}}$ .

The two inference steps described above are graphically depicted in Figure 1c. This is exactly the same setting suggested by Kingma et al. [2014], but here we wish to investigate its applicability in a non-parametric, Gaussian process based framework. The very encouraging results reported below point towards the future direction of applying this technique in the framework of deep Gaussian processes [Damianou and Lawrence, 2013], so as to be able to compare to [Kingma et al., 2014] who considered deep, generative (but nevertheless parametric) models.

### DEMONSTRATION

We evaluated our semi-supervised GP algorithm in two datasets: firstly, we considered 2000 examples from the USPS handwritten digit database [Hull, 1994]. These examples contained the digits  $\{0, 2, 4, 6\}$  and were split so that 800 instances were used as a test set. From the remaining 1200 instances, we selected various portions to be labelled and the rest to be unlabelled. The experiment was repeated 8 times (each time involving different subsets due to different random seeds), so that we can include errorbars in our plots. Secondly, we considered the oil flow data [Bishop and James, 1993] that consist of 1000, 12 dimen-

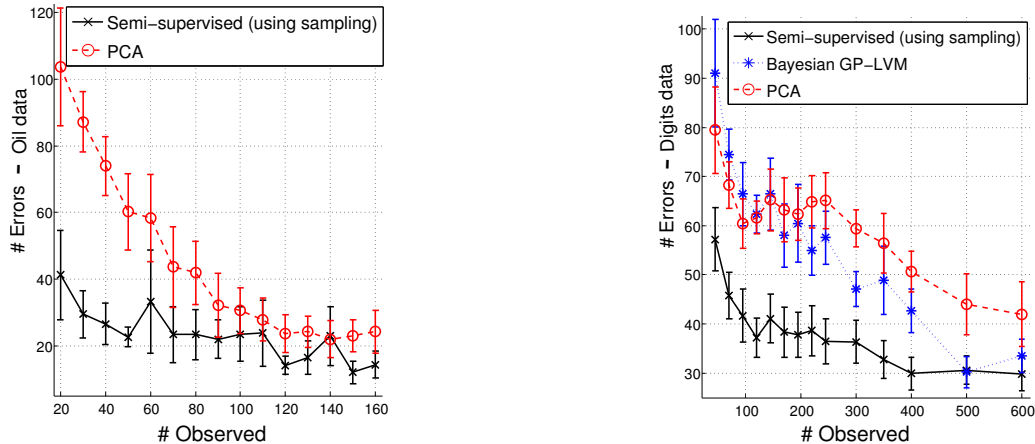


Figure 4: Plots of the number of incorrectly classified test points as a function of  $|\mathbf{Z}^{\mathcal{L}}|$ . Multiple trials were performed, but the resulting errorbars are shown at one standard deviation. In small training sets large errorbars are expected because, occasionally, very challenging instances/outliers can be included and result in high error rates (for all methods) that affect the overall standard deviation. The Bayesian GP-LVM baseline struggled with small training sets and performed very badly in the oil dataset; thus, it is not plotted for clarity.

sional observations belonging to three known classes corresponding to different phases of oil flow. In each of the 10 performed trials, 700 instances were used as a test set whereas the rest were split to different proportions of labelled/unlabelled sets. Multi-label data can also be handled by our method, but this case was not considered here.

Our method learned a low-dimensional embedding  $q(\mathbf{X})$  from all available inputs, and a logistic regression classifier was then trained from the relevant parts of the embedding to the corresponding class space. We experimented with taking different numbers of samples from  $q(\mathbf{X}^{\mathcal{L}})$  for populating the initial labelled set; the difference after increasing over 6 samples was minimal. Also, when using only the mean of  $q(\mathbf{X}^{\mathcal{L}})$  (as opposed to using multiple samples) we obtained worse results (especially in the digits data), but this method still outperformed the baselines. We compared with training the classifier on features learned by (a) a standard Bayesian GP-LVM and (b) PCA, both applied in  $\mathbf{Z}^{\mathcal{L}}$ . Both of the baselines do not take  $\mathbf{Z}^{\mathcal{M}}$  into account, nor do they populate small training sets using sampling. Figure 4 presents results suggesting that our approach manages to effectively take into account unlabelled data. The gain in performance is significant, and our method copes very well even when labelled data is extremely scarce. Notice that all methods would perform better if a more robust classifier was used, but logistic regression was a convenient choice for performing multiple trials fast. Therefore, our conclusions can be safely drawn from the obtained relative errors, since all methods were compared on equal footing.

## 4 DISCUSSION AND FUTURE WORK

We have defined semi-described learning as the scenario where missing and uncertain values occur in the inputs. We

considered semi-described problems to be part of a general class of missing value problems that also includes semi-supervised learning and auto-regressive future state simulation. A principled method for including input uncertainty and partial inputs in Gaussian process models was also introduced to solve these problems within a single, coherent framework. We explicitly represent this uncertainty as approximate posterior distributions which are variationally constrained. This allowed us to further define algorithms for casting the missing value problems as particular instances of learning pipelines which use our variationally constrained GP formulation as a building block. Our algorithms resulted in significant performance improvement in forecasting, regression and classification. We believe that our contribution paves the way for building powerful models for representation learning from real-world, heterogeneous data. In particular, this can be achieved by combining our method with deep Gaussian process models [Damianou and Lawrence, 2013] that use relevance determination techniques [Damianou et al., 2012], so as to consolidate semi-described hierarchies of features that are gradually abstracted to concepts. We plan to investigate the application of these models in settings where control [Deisenroth et al., 2014] or robotic systems learn by simulating future states in an auto-regressive manner and by using incomplete data with minimal human intervention. Transfer learning is another promising direction for applying these models.

## ACKNOWLEDGEMENTS

This research was funded by the European research project EU FP7-ICT (Project Ref 612139 “WYSIWYD”). We thank Michalis Titsias for useful discussions.

## References

- C. M. Bishop and G. D. James. Analysis of multiphase flows using dual-energy gamma densitometry and neural networks. *Nuclear Instruments and Methods in Physics Research*, A327:580–593, 1993.
- O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-supervised Learning*. MIT Press, Cambridge, MA, 2006.
- Z. Dai, A. Damianou, J. Hensman, and N. Lawrence. Gaussian process models with parallelization and GPU acceleration. *arXiv preprint arXiv:1410.4984*, 2014.
- P. Dallaire, C. Besse, and B. Chaïb-Draa. Learning Gaussian process models from uncertain data. In *Neural Information Processing*, pages 433–440. Springer, 2009.
- A. Damianou and N. Lawrence. Deep Gaussian processes. In *Proceedings of the Sixteenth International Workshop on Artificial Intelligence and Statistics (AISTATS)*, pages 207–215. JMLR W&CP 31, 2013.
- A. Damianou and N. Lawrence. Uncertainty propagation in Gaussian process pipelines. *NIPS workshop on modern non-parametrics*, 2014.
- A. Damianou, M. Titsias, and N. D. Lawrence. Variational Gaussian process dynamical systems. In *Advances in Neural Information Processing Systems 24*, pages 2510–2518. 2011.
- A. Damianou, C. Ek, M. Titsias, and N. Lawrence. Manifold relevance determination. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*, pages 145–152. Omnipress, 2012.
- M. P. Deisenroth, R. D. Turner, M. F. Huber, U. D. Hanebeck, and C. E. Rasmussen. Robust filtering and smoothing with Gaussian processes. *Automatic Control, IEEE Transactions on*, 57(7):1865–1871, 2012.
- M. P. Deisenroth, D. Fox, and C. E. Rasmussen. Gaussian processes for data-efficient learning in robotics and control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 99:1, 2014. ISSN 0162-8828.
- C. H. Ek, J. Rihan, P. Torr, G. Rogez, and N. D. Lawrence. Ambiguity modeling in latent spaces. In A. Popescu-Belis and R. Stiefelwagen, editors, *Machine Learning for Multimodal Interaction (MLMI 2008)*, LNCS, pages 62–73. Springer-Verlag, 28–30 June 2008.
- Y. Gal, M. van der Wilk, and C. E. Rasmussen. Distributed variational inference in sparse Gaussian process regression and latent variable models. *arXiv:1402.1389*, 2014.
- Z. Ghahramani and M. I. Jordan. Learning from incomplete data. Technical Report CBCL 108, Massachusetts Institute of Technology, 1994.
- A. Girard, C. E. Rasmussen, J. Quiñero Candela, and R. Murray-Smith. Gaussian process priors with uncertain inputs—application to multiple-step ahead time series forecasting. In *Advances in Neural Information Processing Systems*, pages 529–536, 2003.
- J. J. Hull. A database for handwritten text recognition research. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16:550–554, 1994.
- D. P. Kingma, D. J. Rezende, S. Mohamed, and M. Welling. Semi-supervised learning with deep generative models. *CoRR*, abs/1406.5298, 2014.
- N. D. Lawrence. The Gaussian process latent variable model. Technical Report CS-06-03, The University of Sheffield, Department of Computer Science, 2006.
- N. D. Lawrence and M. I. Jordan. Semi-supervised learning via Gaussian processes. In L. Saul, Y. Weiss, and L. Boutou, editors, *Advances in Neural Information Processing Systems*, volume 17, pages 753–760. Cambridge, MA, 2005. MIT Press.
- N. D. Lawrence and J. Quiñero Candela. Local distance preservation in the GP-LVM through back constraints. In W. Cohen and A. Moore, editors, *Proceedings of the International Conference in Machine Learning*, volume 23, pages 513–520. Omnipress, 2006. ISBN 1-59593-383-2. doi: 10.1145/1143844.1143909.
- A. McHutchon and C. E. Rasmussen. Gaussian process training with input noise. In *NIPS*, 2011.
- J. Oakley and A. O’Hagan. Bayesian inference for the uncertainty distribution of computer model outputs. *Biometrika*, 89(4):769–784, 2002.
- M. Osborne and S. J. Roberts. Gaussian processes for prediction. Technical report, Department of Engineering Science, University of Oxford, 2007.
- J. Quiñero-Candela. *Learning with uncertainty-Gaussian processes and relevance vector machines*. PhD thesis, Technical University of Denmark, 2004.
- J. Quiñero-Cañdela and S. Roweis. Data imputation and robust training with Gaussian processes. *NIPS*, 2003.
- J. Quiñero-Candela, A. Girard, J. Larsen, and C. E. Rasmussen. Propagation of uncertainty in bayesian kernel models-application to multiple-step ahead forecasting. In *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP’03). 2003 IEEE International Conference on*, volume 2, pages II–701. IEEE, 2003.
- C. Rosenberg, M. Hebert, and H. Schneiderman. Semi-supervised self-training of object detection models. In *Application of Computer Vision, 2005. WACV/MOTIONS ’05 Volume 1.*, volume 1, pages 29–36, Jan 2005. doi: 10.1109/ACVMOT.2005.107.
- D. B. Rubin. *Multiple imputation for nonresponse in surveys*, volume 81. John Wiley & Sons, 2004.
- M. Titsias and N. D. Lawrence. Bayesian Gaussian process latent variable model. *Journal of Machine Learning Research - Proceedings Track*, 9:844–851, 2010.
- J. Zhu, A. Ahmed, and E. P. Xing. Medlda: maximum margin supervised topic models. *The Journal of Machine Learning Research*, 13(1):2237–2278, 2012.

---

# Budget Constraints in Prediction Markets

---

**Nikhil Devanur**  
Microsoft Research

**Miroslav Dudík**  
Microsoft Research

**Zhiyi Huang**  
University of Hong Kong

**David M. Pennock**  
Microsoft Research

## Abstract

We give a detailed characterization of optimal trades under budget constraints in a prediction market with a cost-function-based automated market maker. We study how the budget constraints of individual traders affect their ability to impact the market price. As a concrete application of our characterization, we give sufficient conditions for a property we call *budget additivity*: two traders with budgets  $B$  and  $B'$  and the same beliefs would have a combined impact equal to a single trader with budget  $B + B'$ . That way, even if a single trader cannot move the market much, a crowd of like-minded traders can have the same desired effect. When the set of payoff vectors associated with outcomes, with coordinates corresponding to securities, is affinely independent, we obtain that a generalization of the heavily-used logarithmic market scoring rule is budget additive, but the quadratic market scoring rule is not. Our results may be used both descriptively, to understand if a particular market maker is affected by budget constraints or not, and prescriptively, as a recipe to construct markets.

## 1 INTRODUCTION

A prediction market is a central clearinghouse for people with differing opinions about the likelihood of an event—say Hillary Clinton to win the 2016 U.S. Presidential election—to trade monetary stakes in the outcome with one another. At equilibrium, the price to buy a contract paying \$1 if Clinton wins reflects a consensus of sorts on the probability of the event. At that price, and given the wagers already placed, no agent is willing to push the price further up or down. Prediction markets have a good track record of forecast accuracy in many domains [11, 19].

The design of *combinatorial* markets spanning multiple

logically-related events raises many interesting questions. What information can be elicited—the full probability distribution, or specific properties of the distribution? What securities can the market allow traders to buy and sell? How can the market support and ensure a variety of trades? For example, in addition to the likelihood of Clinton winning the election, we may want to elicit information about the distribution of her electoral votes.<sup>1</sup> If we create one security for each possible outcome between 0 and 538, each paying \$1 iff Clinton gets exactly that many electoral votes, the market is called *complete*, allowing us to elicit a full probability distribution. Alternatively, if we create just two securities, one paying out  $\$x$  if Clinton wins  $x$  electoral votes, and the other paying out  $\$x^2$ , we cannot elicit a full distribution, but we can still elicit the mean and variance of the number of electoral votes.

When agents are constrained in how much they can trade only by risk aversion, prediction market prices can be interpreted as a weighted average of traders' beliefs [2, 20], a natural reflection of the “wisdom of the crowd” with a good empirical track record [14] and theoretical support [2]. However, when agents are budget constrained, discontinuities and idiosyncratic results can arise [7, 16] that call into question whether the equilibrium prices can be trusted to reflect any kind of useful aggregation.

We consider prediction markets with an automated market maker [1, 4, 13] that maintains standing offers to trade every security at some price. Unlike a peer-to-peer exchange, all transactions route through the market maker. The common market makers have bounded loss and are (myopically) incentive compatible: the best (immediate) strategy is for a trader to move the market prices of all securities to equal his own belief. The design of such an automated market maker boils down to choosing a convex *cost function* [1]. This amount of design freedom presents an opportunity to seek cost functions that satisfy additional desiderata such as computational tractability [1, 6].

---

<sup>1</sup> A U.S. Presidential candidate receives a number of electoral votes between 0 and 538. The candidate who receives a plurality of electoral votes wins the election.



Most of the literature assumes either risk-neutral or risk-averse traders with unbounded budgets. In this paper, we consider how agents with budget constraints trade in such markets, a practical reality in almost all prediction markets denominated in both real and virtual currencies. Our results help with a systematic study of the market’s *liquidity parameter*, or the parameter controlling the sensitivity of prices to trading volume. Setting the liquidity is a nearly universal practical concern and, at present, is more (black) art than science. We adopt the notion of the “natural budget constraint” introduced by Fortnow and Sami [8]: the agent is allowed only those trades for which the maximum loss for any possible outcome does not exceed the budget.

The main contribution of this paper is a rich, geometric characterization of the impact of budget constraints. Price vectors, outcomes and trader beliefs are embedded in the space of the same dimension as the number of securities. Outcome vectors enumerate security payoffs; belief vectors enumerate the traders’ expectations of payoffs. We consider, for a fixed belief, the locus of the resulting price vectors of an optimal trade as a function of the budget. We show that the price vector moves in the convex hull of the belief and the set of tight outcomes, in a direction that is perpendicular to the set of tight outcomes. We also introduce the concept of *budget additivity*: two agents with budgets  $B$  and  $B'$  and the same beliefs have the same power to move the prices as a single agent with the same belief and budget  $B+B'$ . An absence of budget additivity points to an inefficiency in incorporating information from the traders. We show that budget additivity is a non-trivial property by giving examples of market makers that do not satisfy budget additivity. We give a set of sufficient conditions on the market maker and the set of securities offered which guarantee budget additivity. Further, for two of the most commonly used market makers (the quadratic and logarithmic market scoring rules), we show sufficient conditions on the set of securities that guarantee budget additivity.

Of greatest practical interest is the application of our results to markets consisting of several independent questions, with each question priced according to a separate logarithmic market scoring rule. This setup constitutes a de facto industry standard, and the companies that use (or used) it include Inkling Markets,<sup>2</sup> Consensus Point,<sup>3</sup> Microsoft and Yahoo! [17]. Our Theorems 5.6 and 5.8 show that these markets are budget additive.

Previously, Fortnow and Sami [8] considered a different question: do budget-constrained bidders always move the market prices in the direction of their beliefs? They showed that the answer to this is no: there always exist market prices, beliefs and budgets such that the direction of price movement is not towards the belief. We give a richer char-

acterization of how the market prices move in the presence of budget constraints, by charting the path the prices take with increasing budgets. The impossibility result of Fortnow and Sami [8] can be easily derived from our characterization (see Appendix D).<sup>4</sup>

A designer of a prediction market has a lot of freedom but little guidance, and our results can be used both descriptively and prescriptively. As a descriptive tool, our results enable us to analyze commonly used market makers and understand if budget constraints hamper information aggregation in these markets. As a prescriptive tool, our results can be used to construct markets that are budget additive. In particular, we speculate that budget additivity simplifies the choice of the liquidity parameter in the markets, because it allows considering trader budgets in aggregate.

**Proof overview and techniques.** Our analysis borrows heavily from techniques in convex analysis and builds on the notion of Bregman divergence. We use the special case of Euclidean distance (corresponding to a quadratic market scoring rule) to form our geometric intuition which we then extend to arbitrary Bregman divergences. For the sake of an example, consider a complete market over a finite set of outcomes, where the market prices lie in a simplex, exactly coinciding with the set of probability distributions over outcomes. Every possible outcome imposes a constraint on the set of prices to which a trader can move the market, because the trader is not allowed to exceed the budget if that outcome occurs. The prices satisfying this constraint form a ball with the outcome at its center. The set of feasible prices to which the trader can move the market is therefore the intersection of these balls (see Figure 1).

The key structural result we obtain is the chart of the price movement. Suppose that there is an infinite sequence of agents with infinitesimally small budgets all with the same belief. What is the path along which the prices move from some initial values? This is determined by the agents’ belief and the set of budget constraints that are tight at any point, corresponding to the highest risk outcomes (outcomes with the highest potential loss). We show that the price vector can always be written as a convex combination of these highest risk outcomes and the agents’ belief. Further, the market prices move in a direction that is perpendicular to the affine space of these outcomes.

The agents’ belief partitions the simplex interior into regions, where each region is the interior of the convex hull of the agent belief and a particular subset of outcomes. For a region that is full-dimensional, every interior point can be uniquely written as a convex combination of the agent belief and all except one outcome. Assume that the current price vector lies in this region. In the anticipation of the further development, we call this outcome *profitable* and others *risky*. Motivated by the characterization above, we

<sup>2</sup>[inklingmarkets.com](http://inklingmarkets.com)

<sup>3</sup>[www.consensuspoint.com](http://www.consensuspoint.com)

<sup>4</sup>The full version of this paper on arXiv includes the appendix.

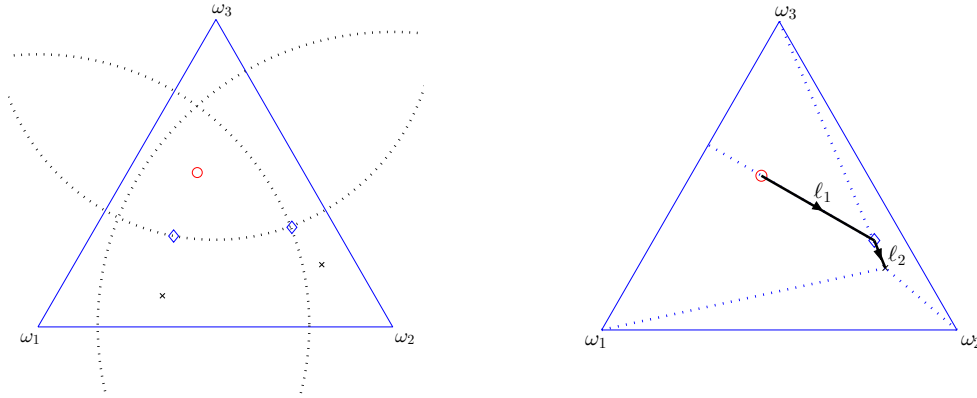


Figure 1: *Left*:  $\circ$  —current state,  $\times$  —belief,  $\diamond$  —optimal action for a given belief and budget. Three circles bound the allowed final states for budget 0.1. We plot optimal actions for two different beliefs. *Right*: A path from the initial state to the belief, consisting of optimal actions for increasing budgets.

move perpendicular to the risky outcomes in the direction towards the agents’ belief. As a result, we increase the risk of risky outcomes (equally for all outcomes), while getting closer to the one profitable outcome (and hence increasing its profit). The characterization then guarantees that the prices along this path are indeed those chosen by traders at increasing budgets, because the risky outcomes yield tight constraints.

We would like the same to be true for the lower dimensional regions as well; that is, for the set of tight constraints to be exactly the corresponding set of outcomes defining the convex hull. In fact, this property is sufficient to guarantee budget additivity. The markets for which the tight constraints are exactly the minimal set of outcomes that define the region the price lies in are budget additive. (We conjecture that the converse holds as well.) The entire path is then as follows: w.l.o.g. you start at a full-dimensional region, move along the perpendicular until you hit the boundary of the region and you are in a lower-dimensional region, move along the perpendicular in this lower-dimensional region, and so on until you reach the belief (see Figure 1). The set of tight constraints is monotonically decreasing. We show that such markets are characterized by a certain *acute angles assumption* on the set of possible outcomes. Loosely speaking, this assumption guarantees that outcomes outside the minimal set behave as the profitable outcome in the above example.

**Other related work.** There is a rich literature on scoring rules and prediction markets. Two of the most studied scoring rules are the quadratic scoring rule [3] and the logarithmic market scoring rule [13]. We consider cost-function-based prediction markets [4, 12], a fully general class under reasonable assumptions [1, 5]. Their equivalence with proper scoring rules has been implicitly noted by Gneiting and Raftery [10]. Several authors have studied relationships between utility functions and price dynamics in prediction markets, drawing a parallel to online learning [2, 5, 9]. Our analysis touches on the problem of setting the

market maker’s liquidity parameter [15, 17], which determines how (in)sensitive prices are to trading volume. With budget additivity, the market designer can optimize liquidity according to aggregate budgets, without worrying about how budgets are partitioned among traders.

## 2 PRELIMINARIES

**Securities and payoffs.** Consider a probability space with a finite set of outcomes  $\Omega \subseteq \mathbb{R}^n$ . A *security* is a financial instrument whose payoff depends on the realization of an outcome in  $\Omega$ . In other words, the payoff of a security is a random variable of the probability space. We consider trading with  $n$  securities corresponding to  $n$  coordinates of the outcomes  $\omega \in \Omega$ . A security can be traded before the realization is observed with the intention that the price of a security serves as a prediction for the expected payoff, i.e., the expected value of the corresponding coordinate.

**Cost function, prices and utilities.** An *automated market maker* always offers to trade securities, for the right price. In fact the price vector is the current prediction of the market maker for the expectation of  $\omega$ . A cost function based market maker is based on a differentiable convex *cost function*,  $C : \mathbb{R}^n \rightarrow \mathbb{R}$ . It is a scalar function of an  $n$ -dimensional vector  $q \in \mathbb{R}^n$  representing the *number of outstanding shares*<sup>5</sup> for our  $n$  securities. We also refer to  $q$  as the *state* of the market.

The vector of *instantaneous prices* of the securities is simply the gradient of  $C$  at  $q$ , denoted by  $p(q) := \nabla C(q)$ . The prices of securities change continuously as the securities are traded, so it is useful to consider the cost of trading a given quantity of securities. The cost of buying  $\delta \in \mathbb{R}^n$  units of securities (where a negative value corresponds to selling) is determined by the path integral  $\int_{\pi} p(\bar{q}) \cdot d\bar{q} = C(q + \delta) - C(q)$ , where  $\pi$  is any smooth

<sup>5</sup>We allow trading fractions of a security. Negative values correspond to short-selling.

curve from  $q$  to  $q + \delta$ .

When the outcome  $\omega$  is realized, the vector of  $\delta$  units of securities pays off an amount of  $\delta \cdot \omega$ . Thus, the realized utility of a trader whose trade  $\delta$  moved the market state from  $q$  to  $q' = q + \delta$  is

$$U(q', \omega; q) := (q' - q) \cdot \omega - C(q') + C(q) .$$

We make a standard assumption that the maximum achievable utility, which is also the maximum loss of the market maker, is bounded by a finite constant (in Section 4, we introduce a standard approach to check this easily). Let  $\mathcal{M}$  be the convex hull of the payoff vectors,  $\mathcal{M} := \text{conv}(\Omega)$ . It is easy to see that  $\mathcal{M}$  contains exactly the vectors  $\mu \in \mathbb{R}^n$  which can be realized as expected payoffs  $\mathbb{E}[\omega]$  for some probability distribution over  $\Omega$ . For a trader who believes that  $\mathbb{E}[\omega] = \mu$ , the expected utility takes form

$$U(q', \mu; q) := \mathbb{E}[U(q', \omega; q)] = (q' - q) \cdot \mu - C(q') + C(q) .$$

Throughout, we consider a single *myopic* trader who trades as if he were the last to trade. A key property satisfied by expected utility is *path independence*: for any  $q, \bar{q}, q' \in \mathbb{R}^n$ ,  $U(q', \mu; \bar{q}) + U(\bar{q}, \mu; q) = U(q', \mu; q)$ , that is, risk-neutral traders have no incentive to split their trades. For a risk-neutral trader,  $q' \in \mathbb{R}^n$  is an optimal action if and only if  $\mu = \nabla C(q') = p(q')$  (this follows from the first-order optimality conditions). In other words, the trader is incentivized to move the market to the prices corresponding to his belief as long as such prices exist. In general, there may be multiple states yielding the same prices, so the inverse map  $p^{-1}(\mu)$  returns a *set*, which can be empty if no state yields the price vector  $\mu$ .

Commonly-used cost functions include the quadratic cost, logarithmic market-scoring rule (LSMR) and the log-partition function. They are described in detail in Appendix A. The quadratic cost is defined by  $C(q) = \frac{1}{2} \|q\|_2^2$  and  $p(q) = q$ . Log-partition function is defined as  $C(q) = \ln(\sum_{\omega \in \Omega} e^{q \cdot \omega})$ . It subsumes LSMR as a special case for the *complete market* with the outcomes corresponding to vertices of the simplex. The prices under log-partition cost correspond to the expected value of  $\omega$  under the distribution  $P_q(\omega) = e^{q \cdot \omega - C(q)}$  over  $\Omega$ , i.e.,  $p(q) = \mathbb{E}_{P_q}[\omega]$ .

**Budget constraints.** Trading in prediction markets needs an investment of capital. It is possible that an agent loses money on the trade, in particular  $U(q', \omega; q)$  could be negative for some  $\omega$ . One restriction on how an agent trades could be that he is unable to sustain a big loss, due to a budget constraint. We consider the notion of *natural budget constraint* defined by Fortnow and Sami [8] which states that the loss of the agent is at most his budget, for all  $\omega \in \Omega$ . Given a starting market state  $q_0$  and a budget of  $B \geq 0$ , a trader with the belief  $\mu \in \mathcal{M}$  then solves the problem:

$$\begin{aligned} \max_{q \in \mathbb{R}^n} & U(q, \mu; q_0) \\ \text{s.t.} & U(q, \omega; q_0) \geq -B \quad \forall \omega \in \Omega . \end{aligned} \quad (2.1)$$

For quadratic costs, each constraint corresponds to a sphere with one of the outcomes at its center, so the feasible region is an intersection of these spheres. We will later see that this generalizes to an intersection of balls w.r.t. a Bregman divergence for general costs.

In general, there may be multiple  $q$  optimizing this objective. In the following definition we introduce notation for various solution sets we will be analyzing. The belief  $\mu$  is fixed throughout most of the discussion, so we suppress the dependence on  $\mu$ .

**Definition 2.1** (Solution sets). Let  $\hat{Q}(B; q_0)$  denote the set of solutions of Convex Program (2.1) for a fixed initial state and budget. Let  $\hat{Q}(q_0) = \bigcup_{B \geq 0} \hat{Q}(B; q_0)$  denote the set of solutions of (2.1) for a fixed initial state across all budgets. Let  $\hat{Q}(\nu; q_0) = p^{-1}(\nu) \cap \hat{Q}(q_0)$  denote the set of states  $q$  that optimize (2.1) for some budget  $B$  and yield the market price vector  $\nu$ .

The next theorem shows that solutions for a fixed initial state and budget always yield the same price vector. It is proved in Appendix B.

**Theorem 2.2.** *If  $q, q' \in \hat{Q}(B; q_0)$ , then  $p(q) = p(q')$ .*

**Geometry of linear spaces.** We finish this section by reviewing a few standard geometric definitions we use in next sections. Let  $X \subseteq \mathbb{R}^n$ . Then  $\text{aff}(X)$  denotes the *affine hull* of the set  $X$  (i.e., the smallest affine space including  $X$ ). We write  $X^\perp$  to denote the *orthogonal complement* of  $X$ :  $X^\perp := \{u \in \mathbb{R}^n : u \cdot (x' - x) = 0 \text{ for all } x, x' \in X\}$ . We use the convention  $\emptyset^\perp = \mathbb{R}^n$ . A set  $\mathcal{K} \in \mathbb{R}^n$  is called a *cone* if it is closed under multiplication by positive scalars. If a cone is convex, it is also closed under addition. Since  $\Omega$  is finite, the realizable set  $\mathcal{M} = \text{conv}(\Omega)$  is a polytope. Its boundary can be decomposed into *faces*. More precisely,  $X \subseteq \Omega$ ,  $X \neq \emptyset$ , forms a *face* of  $\mathcal{M}$  if  $X$  is the set of maximizers over  $\Omega$  of some linear function.<sup>6</sup> We also view  $X = \emptyset$  as a face of  $\mathcal{M}$ . With this definition, for any two faces  $X, X'$ , also their intersection  $X \cap X'$  is a face.

### 3 CHARACTERIZING SOLUTION SETS

We start with the optimality (KKT) conditions for the Convex Program (2.1), as characterized by the next lemma. One of the key conditions is that the solution prices must be in the convex hull of the belief  $\mu$  and all the  $\omega$ 's for which the budget constraints are *tight*. The set of tight constraints is always a face of the polytope  $\mathcal{M}$ . We allow an empty set as a face, which corresponds to the case when none of the constraints are tight and the solution prices coincide with  $\mu$ . The proof follows by analyzing KKT conditions (see Appendix C of the full version for details).

<sup>6</sup>Strictly speaking, this is the definition of an *exposed face*, but all faces of a polytope are exposed, so the distinction does not matter here. The exposed face is typically defined to be  $\text{conv}(X)$ , but in this paper, it is more convenient to work with  $X$  directly.

**Lemma 3.1** (KKT lemma). *Let  $q_0 \in \mathbb{R}^n$ . Then  $q \in \hat{Q}(B; q_0)$  if and only if there exists a face  $X \subseteq \Omega$  such that the following conditions hold:*

- (a)  $U(q, x; q_0) = U(q, x'; q_0)$ , or equivalently  $(q - q_0) \cdot (x' - x) = 0$ , for all  $x, x' \in X$
- (b)  $U(q, \omega; q_0) \geq U(q, x; q_0)$ , or equivalently  $(q - q_0) \cdot (\omega - x) \geq 0$ , for all  $x \in X, \omega \in \Omega \setminus X$
- (c)  $p(q) \in \text{conv}(X \cup \{\mu\})$
- (d)  $B = -U(q, x; q_0)$  for all  $x \in X$  if  $X \neq \emptyset$ , or  $B \geq \max_{\omega \in \Omega} [-U(q, \omega; q_0)]$  if  $X = \emptyset$

where conditions (a) and (b) hold vacuously for  $X = \emptyset$ .

The condition (a) requires that  $q - q_0$  be orthogonal to the active set  $X$ . The set of points satisfying conditions (a) and (c) will be called the Bregman perpendicular and will be defined in the next section. The condition (b) is a statement about acuteness of the angle between  $q - q_0$  (the perpendicular) and the outcomes. It will be the basis of our *acute angles* assumption. The condition (d) just states how the budget is related to the active set  $X$ .

**Witness cones and minimal faces.** We now introduce some notation to help us state reinterpretations of the conditions in Lemma 3.1. First of all, given a face  $X$ , what is the set of  $q$ 's that satisfy conditions (a) and (b)? This is captured by what we call the *witness cone*.

**Definition 3.2.** The *witness cone* for a face  $X \subseteq \Omega$  is defined as  $\mathcal{K}(X) := \{u \in \mathbb{R}^n : u \cdot (\omega - x) \geq 0 \text{ for all } x \in X, \omega \in \Omega\}$  if  $X \neq \emptyset$ , and  $\mathcal{K}(X) := \mathbb{R}^n$  if  $X = \emptyset$ .

The following two properties of witness cones are immediate from the definition:

- *Anti-monotonicity:* if  $X \subseteq X'$ , then  $\mathcal{K}(X) \supseteq \mathcal{K}(X')$ .
- *Orthogonality:*  $\mathcal{K}(X) \subseteq X^\perp$ .

A state  $q$  satisfies conditions (a) and (b) for a given face  $X$  if and only if  $q - q_0 \in \mathcal{K}(X)$ . Now given a state  $q$ , consider the set of faces that could satisfy condition (c). This set has a useful structure, namely that there is a unique minimal face (proved in Appendix C of the full version).

**Definition 3.3.** Given a price vector  $\nu \in \mathcal{M}$ , the *minimal face* for  $\nu$  is the minimal face  $X$  (under inclusion) s.t.  $\nu \in \text{conv}(X \cup \{\mu\})$ . The minimal face for  $\nu$  is denoted as  $X_\nu$ .

With the existence of a minimal face and the anti-monotonicity of the witness sets, it follows that if  $q$  and  $X$  satisfy conditions (a), (b) and (c), then so do  $q$  and  $X_{p(q)}$ . Thus we obtain the following version of Lemma 3.1 (proved in Appendix C of the full version).

**Theorem 3.4** (Characterization of Solution Sets).  $q \in \hat{Q}(q_0)$  if and only if  $q \in [q_0 + \mathcal{K}(X_{p(q)})]$ .

Using Theorem 3.4, we immediately obtain a characterization of when a price vector  $\nu$  could be the price vector of an optimal solution to (2.1).

**Corollary 3.5.**  $\hat{Q}(\nu; q_0) = p^{-1}(\nu) \cap [q_0 + \mathcal{K}(X_\nu)]$ . In particular,  $\nu$  is the price vector of an optimal solution to (2.1) if and only if  $p^{-1}(\nu) \cap [q_0 + \mathcal{K}(X_\nu)] \neq \emptyset$ .

We now study an example using the above characterization. More examples can be found in Appendix E of the full version.

**Example 3.6** (Quadratic cost on an obtuse triangle; see Example E.2 in the full version for details). Consider the following outcome space, belief, and the sequence of market states (depicted in Figure 2):

$$\begin{aligned} \omega_1 &= (0.0, 0.0) & q_0 = \nu_0 &= \frac{11}{14}\omega_2 + \frac{3}{14}\omega_3 \\ \omega_2 &= (1.8, 0.0) & q_1 = \nu_1 &= \frac{1}{3}\omega_2 + \frac{2}{3}\mu \\ \omega_3 &= (6.0, 4.2) & q_2 = \nu_2 &= \frac{1}{9}\omega_1 + \frac{8}{9}\mu \\ \mu = q_\mu &= (2.7, 1.8) & q_3 = \nu_3 &\approx \frac{1}{19}\omega_1 + \frac{18}{19}\mu \end{aligned}$$

Using the KKT lemma, we can show for  $j = 1, 2, 3$ , that  $q_j = \nu_j$  is an optimal action at  $q_{j-1} = \nu_{j-1}$  under belief  $\mu$ , with the corresponding budgets as:

|                        | $\omega_1$ | $\omega_2$ | $\omega_3$ |                    |
|------------------------|------------|------------|------------|--------------------|
| $U(q_1, \cdot; q_0)$   | 0.45       | -0.09      | -0.09      | $B_{01} = 0.09$    |
| $U(q_2, \cdot; q_1)$   | -0.56      | -0.56      | 1.12       | $B_{12} = 0.56$    |
| $U(q_3, \cdot; q_2)$   | -0.565     | -0.28      | 0.82       | $B_{23} = 0.565$   |
| $U(q_\mu, \cdot; q_0)$ | -1.215     | -1.215     | 2.565      | $B_{0\mu} = 1.215$ |

The above table also shows that the budget  $B_{0\mu} = 1.215$  suffices to move directly from  $q_0$  to  $q_\mu$ . However, note that the sum  $B_{01} + B_{12} + B_{23} = 1.215 = B_{0\mu}$ , but  $\nu_3 \neq \mu$ , i.e., after the sequence of optimal actions with budgets  $B_{01}$ ,  $B_{12}$ , and  $B_{23}$ , the market is still not at the belief shared by all agents, even though with the budget  $B_{0\mu}$ , it would have reached it.

**Budget additivity.** The above example suggests that multiple traders with the same belief may have less power in moving the market state towards their belief compared to a single trader with the same belief and the combined budget. Since prediction markets aim to efficiently aggregate information from agents, it is natural to ask under what conditions multiple traders with the same beliefs do have a combined impact equal to a single trader with the combined budget.

Next, we formally define this property as *budget additivity*. We then define the Euclidean version of the acute angles condition that we show is sufficient for budget additivity.

**Definition 3.7** (Budget additivity). We say that a prediction market is budget additive on  $\mathcal{M}' \subseteq \mathcal{M}$  if for all beliefs  $\mu \in \mathcal{M}'$  and all initial states  $q_0 \in p^{-1}(\mathcal{M}')$  the following holds: For any budgets  $B, B' \geq 0$  and any sequence of solutions  $q \in \hat{Q}(B; q_0)$  and  $q' \in \hat{Q}(B'; q)$ , we have  $p(q), p(q') \in \mathcal{M}'$  and  $q' \in \hat{Q}(B + B'; q_0)$ .

In other words, the market is budget additive if the sequence of optimal actions of two agents with the same be-

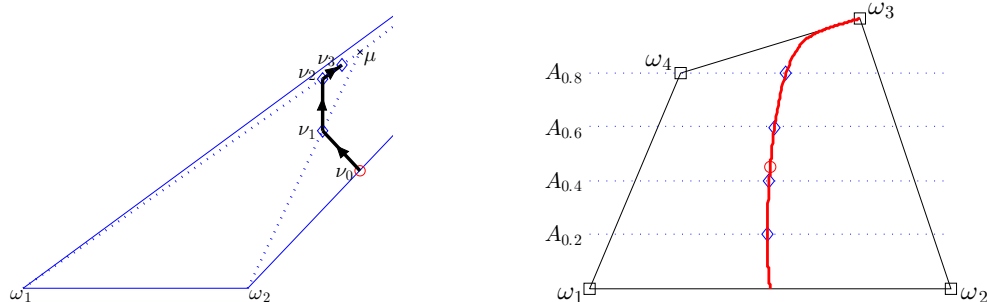


Figure 2: *Left*: An example of non-additive budgets when payoffs form obtuse angles (see Example 3.6 and its extended version Example E.2 in the full version). *Right*: An examples of a non-linear perpendicular for the log-partition cost.

lief and budgets  $B$  and  $B'$  is also an optimal action of a single agent with the same belief and a larger budget  $B + B'$ . Thanks to Theorem 2.2 we then also obtain that the price vector following the sequence of optimal actions by the two agents is the same as the price vector after the optimal action by an agent with the combined budget (all with the same beliefs).

We now state the acute angles assumption for the Euclidean case, to give an intuition. Our acute angles assumption (Definition 5.1) is a generalization of this. We later show that the acute angles property is sufficient for budget additivity (Theorem 5.2).

**Definition 3.8.** We say that the *Euclidean acute angles* hold for a face  $X$ , if the angle between any point  $\bar{\nu} \in \mathcal{M}$ , its projection on the affine hull of  $X$  and any payoff  $\omega \in \Omega$  is non-obtuse (the angle is measured at the projection).

Based on the above example, one may hypothesize that the obtuse angles are to blame for the lack of budget additivity. In the following sections we will show that this is indeed the case, but that the notion of obtuse/acute angles depends on the Bregman divergence. In particular, the above example would have been budget-additive if we used the log-partition cost instead of the quadratic cost.

## 4 BREGMAN DIVERGENCE AND PERPENDICULARS

We will see next that the utility function  $U$  can be written as the difference of two terms measuring the distance between the belief and the market state before and after the trade. This distance measure is the mixed *Bregman divergence*.<sup>7</sup> To define the Bregman divergence, first let  $C^* : \mathbb{R}^n \rightarrow (-\infty, \infty]$  be the convex conjugate of  $C$  defined as  $C^*(\nu) := \sup_{q' \in \mathbb{R}^n} [q' \cdot \nu - C(q')]$ . Since  $C^*$  is a supremum of linear functions, it is convex lower semi-continuous. Up to a constant, it characterizes the maximum achievable utility on an outcome  $\omega$  for a fixed initial state  $q$

<sup>7</sup>Our notion of Bregman divergence is more general than typically assumed in the literature.

as  $\sup_{q' \in \mathbb{R}^n} U(q', \omega; q) = C^*(\omega) + [C(q) - q \cdot \omega]$ . The term in the brackets is always finite, but  $C^*$  might be positive infinite. We make a standard assumption that  $C^*(\omega) < \infty$  for all  $\omega \in \Omega$ , i.e., that the maximum achievable utility, which is also the maximum loss of the market maker, is bounded by a finite constant. By convexity, this implies that  $C^*(\mu) < \infty$  for all  $\mu \in \mathcal{M}$ . The *Bregman divergence* derived from  $C$  is a function  $D : \mathbb{R}^n \times \mathbb{R}^n \rightarrow (-\infty, \infty]$  measuring the maximum expected utility under belief  $\mu$  at a state  $q$

$$D(q, \mu) := C(q) + C^*(\mu) - q \cdot \mu = \sup_{q' \in \mathbb{R}^n} U(q', \mu; q) .$$

From the convexity of  $C$  and  $C^*$  and the definition of  $C^*$ , it is clear that: (i)  $D$  is convex and lower semi-continuous in each argument separately; (ii)  $D$  is non-negative; and (iii)  $D$  is zero iff  $p(q) = \nabla C(q) = \mu$ . By the bounded loss assumption, Bregman divergence is finite on  $\mu \in \mathcal{M}$ . For  $\mu \in \mathcal{M}$ , we can write

$$U(q', \mu; q) = D(q, \mu) - D(q', \mu) . \quad (4.1)$$

Thus, maximizing the expected utility is the same as minimizing the Bregman divergence between the state  $q'$  and the belief  $\mu$ . From Eq. (4.1) it is also clear that each constraint in (2.1) is equivalent to  $D(q, \omega) \leq D(q_0, \omega) + B$ , and the geometric interpretation is that the agent seeks to find the state closest to his belief, within the intersection of Bregman balls

For the quadratic cost, we have  $C^*(\nu) = \frac{1}{2} \|\nu\|^2$  and  $D(q, \nu) = \frac{1}{2} \|q - \nu\|^2$ , i.e., the Bregman divergence coincides with the Euclidean distance squared. For log-partition cost, we have  $C^*(\nu) = \sum_{\omega \in \Omega} P_\nu(\omega) \ln P_\nu(\omega)$  where  $P_\nu$  is the distribution maximizing entropy among  $P$  satisfying  $\mathbb{E}_P[\omega] = \nu$ . The Bregman divergence is the KL-divergence between  $P_q$  and  $P_\nu$ :  $D(q, \nu) = \text{KL}(P_\nu \| P_q)$ .

**Convex analysis.** We overview a few standard definitions and results from convex analysis. For  $X \subseteq \mathbb{R}^n$ , we write  $\text{ri} X$  for the *relative interior* of  $X$  (i.e., the interior relative to the affine hull). For a convex function  $F : \mathbb{R}^n \rightarrow (-\infty, \infty]$ , we define its *effective domain*

as  $\text{dom } F := \{u \in \mathbb{R}^n : F(u) < \infty\}$  (i.e., the set of points where it is finite). The *subdifferential* of  $F$  at a point  $u$  is the set  $\partial F(u) := \{v \in \mathbb{R}^n : F(u') \geq F(u) + (u' - u) \cdot v \text{ for all } u' \in \mathbb{R}^n\}$ . We say that  $F$  is *subdifferentiable* at  $u$  if  $\partial F(u) \neq \emptyset$ . A standard result of convex analysis states that  $F$  is always subdifferentiable on a superset of  $\text{ri dom } F$ . If  $F$  is not only convex, but also lower semi-continuous, then  $\partial F$  and  $\partial F^*$  are inverses in the sense that  $v \in \partial F(u)$  iff  $u \in \partial F^*(v)$ . If  $F$  is *differentiable* everywhere on  $\mathbb{R}^n$ , then  $F^*$  is strictly convex on  $\text{ri dom } F^*$ .

Let  $\text{im } p := \{p(q) : q \in \mathbb{R}^n\}$  denote the set of prices that can be expressed by market states. The implications for our setting are that: (i)  $C^*$  is subdifferentiable on  $\text{im } p$ ; (ii)  $p^{-1}(\nu) = \partial C^*(\nu)$  for all  $\nu \in \mathbb{R}^n$ ; (iii) all beliefs in  $\text{ri dom } C^*$  can be expressed by some state  $q$ ; (iv)  $C^*$  is strictly convex on  $\text{ri dom } C^*$ , and similarly  $D(q, \nu)$  is strictly convex on  $\text{ri dom } C^*$  as a function of the second argument.

#### Assumptions on the cost function.

- *Convexity and differentiability on  $\mathbb{R}^n$ .*  $C$  is convex and differentiable on  $\mathbb{R}^n$ .
- *Finite loss.*  $\mathcal{M} \subseteq \text{dom } C^*$ , i.e.,  $C^*$  is finite on  $\mathcal{M}$ .
- *Inclusion of the relative interior.*  $\text{ri } \mathcal{M} \subseteq \text{ri dom } C^*$ .

The first two assumptions are standard. The third assumption is a regularity condition that we require in our results. Here we briefly discuss how it compares with the finite loss assumption. While the two assumptions look similar, neither of them implies the other. For example, if  $\text{dom } C^*$  is an  $n$ -dimensional simplex and  $\mathcal{M}$  is one of its lower dimensional faces, which are lower dimensional simplices, then the finite loss assumption holds, but the inclusion assumption does not. Similarly, for  $n = 1$  and  $\mathcal{M} = [0, 1]$ , the inclusion assumption is satisfied by the conjugate  $C^*(\nu) = 1/\nu + 1/(1 - \nu)$  on  $\nu \in (0, 1)$  and  $C^*(\nu) = \infty$  on  $\nu \notin (0, 1)$ , but this conjugate does not satisfy the finite loss assumption.

We do not view the inclusion assumption as very restrictive, since it is satisfied by many common cost functions. For instance, it always holds when  $C$  is constructed as in [1], because their construction guarantees  $\text{dom } C^* = \mathcal{M}$ . However, the inclusion assumption might not hold for cost functions that allow arbitrage (e.g., [6]).

Our main result relies on strict convexity of  $C^*$  on  $\text{ri dom } C^*$ , so some of our statements will require that the market prices and beliefs lie in that set. The inclusion assumption above guarantees that at the minimum  $\text{ri } \mathcal{M} \subseteq \text{ri dom } C^*$ , but the boundary of  $\mathcal{M}$  is not necessarily included. To allow some generality beyond  $\text{ri } \mathcal{M}$ , we define the set

$$\tilde{\mathcal{M}} := \begin{cases} \mathcal{M} & \text{if } \mathcal{M} \subseteq \text{ri dom } C^* \\ \text{ri } \mathcal{M} & \text{otherwise.} \end{cases}$$

In either case we obtain that  $\tilde{\mathcal{M}} \subseteq \text{ri dom } C^* \subseteq \text{im } p$ , i.e., beliefs in  $\tilde{\mathcal{M}}$  can be expressed by some state  $q$ . For the quadratic cost,  $\tilde{\mathcal{M}} = \mathcal{M}$ . For the log-partition cost,  $\tilde{\mathcal{M}} = \text{ri } \mathcal{M}$ .

**Perpendiculars.** We now define the notion of a Bregman perpendicular to an affine space. This is a *constructive* definition. It plays a central role in the definition of the acute angles assumption, and also in the proof of the main result (Theorem 5.2). We will see that the set of optimal price vectors for different budgets is a sequence of Bregman perpendiculars. Naturally, perpendiculars are closely related to the conditions in Lemma 3.1; in particular to the set of  $q$ 's that satisfy conditions (a) and (c) for a given face  $X$ .

For quadratic costs, Bregman perpendiculars coincide with the usual Euclidean perpendiculars. Consider an affine space and a point not in it. A projection of the point onto the space is the point in the space that is closest in Euclidean distance to the given point. Now consider moving this affine space towards the projected point. The locus of the projection as we move the space is the perpendicular to the space through the given point. We extend this definition to arbitrary Bregman divergences by defining the projection using the corresponding Bregman divergence.

A Bregman perpendicular is determined by three geometric objects within the affine hull  $\text{aff}(\text{dom } C^*)$ . The first of these is an affine space, say  $A_0 \subseteq \text{aff}(\text{dom } C^*)$ . The second is a point  $a_1 \in \text{aff}(\text{dom } C^*) \setminus A_0$ . The affine space  $A = \text{aff}(A_0 \cup \{a_1\}) \subseteq \text{aff}(\text{dom } C^*)$  will be the ambient space that will contain the perpendicular. Define parallel spaces to  $A_0$  in  $A$ , for an arbitrary point  $a_0 \in A_0$ , as  $A_\lambda := A_0 + \lambda(a_1 - a_0)$  for  $\lambda \in \mathbb{R}$ . Note that the definition of  $A_\lambda$  is independent of the choice of  $a_0$ . The third geometric object is a market state  $q \in \mathbb{R}^n$  such that  $p(q) \in A$ . For technical reasons, we will define a perpendicular at  $q$  rather than a more natural notion, which would be at  $p(q)$ . Our reason for switching into  $q$ -space is that inner products, defining optimality of the Bregman projection, are between elements of  $q$ -space and  $\nu$ -space (the two spaces coincide for Euclidean distance). For all  $\lambda \in \mathbb{R}$  define a Bregman projection of  $q$  onto  $A_\lambda$  as

$$\nu_\lambda := \underset{\nu \in A_\lambda}{\text{argmin}} D(q, \nu) .$$

Since  $D(q, \nu)$  is bounded from below and lower semi-continuous, the minimum is always attained (but it may be equal to  $\infty$ ). If it is attained at more than one point, we choose an arbitrary minimizer. Whenever we can choose  $\nu_\lambda \in \text{ri dom } C^*$ , this  $\nu_\lambda$  must be the unique minimizer by strict convexity of  $D(q, \cdot)$  on  $\text{ri dom } C^*$ , and the minimum is finite. We use these  $\nu_\lambda$ 's to define the perpendicular:

**Definition 4.1.** Given  $A_0$ ,  $a_1$  and  $q$  as above, the  $a_1$ -perpendicular to  $A_0$  at  $q$  is a map  $\gamma : \lambda \mapsto \nu_\lambda$  defined over  $\Lambda := \{\lambda \in \mathbb{R} : \nu_\lambda \in \text{ri dom } C^*\}$ . We call  $\Lambda$  the domain of the perpendicular. We define a total order on

$\nu_\lambda, \nu_{\lambda'} \in \text{im } \gamma$  as  $\nu_\lambda \preceq \nu_{\lambda'}$  iff  $\lambda \leq \lambda'$ .

In Appendix F.2 of the full version, we show that perpendiculars are continuous maps. The name perpendicular is justified by the following proposition which matches our Euclidean intuition that the perpendiculars can be obtained by intersecting the ambient space  $A$  with the affine space which passes through  $q$  and is orthogonal to  $A_0$ . It also shows that the perpendicular corresponds to the set of prices that satisfy conditions (a) and (c) with the convex hull relaxed to the affine hull (when  $A_0$  is the affine hull of face  $X$ , point  $a_1$  coincides with  $\mu$  and  $q$  is the initial state). Recall that for an arbitrary set  $X \subseteq \mathbb{R}^n$ , its orthogonal complement is defined as  $X^\perp := \{u : u \cdot (x' - x) = 0 \text{ for all } x, x' \in X\}$ .

**Proposition 4.2.** *Let  $\gamma$  be the  $a_1$ -perpendicular to  $A_0$  at  $q$ , and let  $A = \text{aff}(A_0 \cup \{a_1\})$ . The following two statements are equivalent for any  $\nu' \in \mathbb{R}^n$ :*

- (i)  $\nu' \in \text{im } \gamma$
- (ii)  $\nu' \in A \cap (\text{ri dom } C^*), p^{-1}(\nu') \cap (q + A_0^\perp) \neq \emptyset$

Proposition 4.2 is proved in Appendix F of the full version. The perpendiculars have the following closure property which is useful for showing budget additivity (also proved in Appendix F of the full version):

**Proposition 4.3.** *Under the assumptions of Proposition 4.2,  $\gamma$  is also the  $a_1$ -perpendicular to  $A_0$  at any  $q' \in p^{-1}(\text{im } \gamma) \cap (q + A_0^\perp)$ .*

## 5 BUDGET ADDITIVITY

We now state the *acute angles* property which links the Bregman perpendicular and Corollary 3.5, and is sufficient for budget additivity.

**Definition 5.1.** We say that the *acute angles* hold for a face  $X$ , if for every  $\mu$ -perpendicular  $\gamma$  to  $X$  at  $q$ , such that  $\mu \in \tilde{\mathcal{M}}$  and  $q \in p^{-1}(\tilde{\mathcal{M}})$ , the following holds: If  $\nu' \in \text{im } \gamma$  and  $\nu' \succeq p(q)$ , then  $p^{-1}(\nu') \cap [q + \mathcal{K}(X)] \neq \emptyset$ .

The motivation for the name ‘‘acute angles’’ comes from the Euclidean distance case, where this assumption is equivalent to Definition 3.8 (see Proposition G.1 in the full version). The acute angles property is non-trivial and we have seen that without this property, budget additivity need not hold; we conjecture that it is also a necessary condition. After stating the main theorem, we analyze in more detail when the acute angles are satisfied by the quadratic and log-partition costs.

We now state the main result, that the acute angles are sufficient for budget additivity:

**Theorem 5.2** (Sufficient conditions for budget additivity). *If acute angles hold for every face  $X \subseteq \Omega$ , then the prediction market is budget additive on  $\tilde{\mathcal{M}}$ .*

**Sufficient conditions for acute angles.** We next give the sufficient conditions when the acute angles hold for the quadratic and log-partition cost functions. We also show that the acute angles hold for all one-dimensional outcome spaces, and that they are preserved by taking *direct sums* of markets. Recall that a set  $\mathcal{K} \in \mathbb{R}^n$  is called a cone if it is closed under multiplication by positive scalars. A cone is called *acute*, if  $x \cdot y \geq 0$  for all  $x, y \in \mathcal{K}$ . An *affine cone* with the vertex  $a_0$  is a set  $\mathcal{K}'$  of the form  $a_0 + \mathcal{K}$  where  $\mathcal{K}$  is a cone.

**Theorem 5.3** (Sufficient condition for quadratic cost). *Let  $X$  be a face and  $A'$  be the affine space  $a_0 + X^\perp$  for an arbitrary  $a_0 \in \text{aff}(X)$ . Acute angles hold for the face  $X$  and the quadratic cost if and only if the projection of  $\Omega$  (or, equivalently,  $\mathcal{M}$ ) on  $A'$  is contained in an affine acute cone with the vertex  $a_0$ .*

**Corollary 5.4.** *Acute angles hold for the quadratic cost and a hypercube  $\Omega = \{0, 1\}^n$ .*

**Corollary 5.5.** *Acute angles hold for the quadratic cost and simplex  $\Omega = \{e_i : i \in [n]\}$  where  $[n] = \{1, 2, \dots, n\}$  and  $e_i$  is the  $i$ -th vector of the standard basis in  $\mathbb{R}^n$ .*

**Theorem 5.6** (Log-partition over affinely independent outcomes). *If the set  $\Omega$  is affinely independent then acute angles assumption is satisfied for the log-partition cost.*

**Theorem 5.7** (One-dimensional outcome spaces). *Acute angles hold for any cost function if  $\mathcal{M}$  is a line segment.*

Let  $\Omega_1 \subseteq \mathbb{R}^{n_1}$  and  $\Omega_2 \subseteq \mathbb{R}^{n_2}$  be outcome spaces with costs  $C_1$  and  $C_2$ . We define the *direct sum* of  $\Omega_1$  and  $\Omega_2$  to be the outcome space  $\Omega = \Omega_1 \times \Omega_2$  with the cost  $C : \mathbb{R}^{n_1+n_2} \rightarrow \mathbb{R}$  defined as  $C(q_1, q_2) = C_1(q_1) + C_2(q_2)$ .

**Theorem 5.8** (Acute angles for direct sums). *If acute angles hold for  $\Omega_1$  with cost  $C_1$ , and  $\Omega_2$  with cost  $C_2$ , then they also hold for their direct sum.*

As a direct consequence of this theorem, we obtain that the log-partition cost function satisfies the acute angles assumption on a hypercube. More generally, any direct sum of costs on line segments satisfies the acute angles. This means that all cost-based prediction markets consisting of independent binary questions are budget additive, regardless of costs used to price individual questions.

As mentioned in the introduction, a vast number of deployed cost-based prediction markets consists of independent questions (not necessarily binary), each priced according to an LMSR (i.e., a log-partition cost on a simplex). Theorems 5.6 and 5.8 imply that this industry standard is budget additive.

### 5.1 Proof of Theorem 5.2

In this section we sketch the proof of Theorem 5.2 (for a complete proof see Appendix H of the full version). We proceed in several steps. Let  $\nu_0 = p(q_0)$ . Assuming acute

angles, we begin by constructing an oriented curve  $L$  joining  $\nu_0$  with  $\mu$ , by sequentially choosing portions of perpendiculars for monotonically decreasing active sets. We then show that budget additivity holds for any solutions with prices in  $L$ , and finally show that the curve  $L$  is the locus of the optimal prices of solutions  $\hat{Q}(q_0)$ , as well as optimal prices of solutions  $\hat{Q}(q)$  for any  $q \in \hat{Q}(q_0)$ .

**Part 1: Construction of the solution path  $L$ .** In this part, we construct:

- a sequence of prices  $\nu_0, \nu_1, \dots, \nu_k$  with  $\nu_0 = p(q_0)$  and  $\nu_k = \mu$
- a sequence of oriented curves  $\ell_0, \dots, \ell_{k-1}$  where each  $\ell_i$  goes from  $\nu_i$  to  $\nu_{i+1}$
- a monotone sequence of sets  $\Omega \supseteq X_0 \supset X_1 \supset \dots \supset X_k = \emptyset$ , such that the following *minimality* property holds:  $X_i$  is the minimal face for all  $\nu \in (\text{im } \ell_i) \setminus \{\nu_{i+1}\}$  for  $i \leq k-1$ , and  $X_k$  is the minimal face for  $\nu_k$ .
- a sequence of states  $q_1, \dots, q_{k-1}$  such that  $q_i \in p^{-1}(\nu_i) \cap [q_{i-1} + \mathcal{K}(X_{i-1})]$

The curves  $\ell_i$  will be referred to as *segments*. The curve obtained by concatenating the segments  $\ell_0$  through  $\ell_{k-1}$  will be called the *solution path* and denoted  $L$ . In the special case that  $\nu_0 = \mu$ , we have  $k = 0$ ,  $X_0 = \emptyset$  and  $L$  is a degenerate curve with  $\text{im } L = \{\mu\}$ .

If  $\nu_0 \neq \mu$ , we construct the sequence of segments iteratively. Let  $X_0 \neq \emptyset$  be the minimal face such that  $\nu_0 \in \text{conv}(X_0 \cup \{\mu\})$ . By the minimality,  $\mu \notin \text{aff}(X_0)$ . Let  $\gamma$  be the  $\mu$ -perpendicular to  $\text{aff}(X_0)$  at  $q_0$ . The curve  $\gamma$  passes through  $\nu_0$  and eventually reaches the boundary of  $\text{conv}(X_0 \cup \{\mu\})$  at some  $\nu_1$  by continuity of  $\gamma$  (see Theorem F.3). Let segment  $\ell_0$  be the portion of  $\gamma$  going from  $\nu_0$  to  $\nu_1$ .

This construction gives us the first segment  $\ell_0$ . There are two possibilities:

1.  $\nu_1 = \mu$ ; in this case we are done;
2.  $\nu_1$  lies on a lower-dimensional face of  $\text{conv}(X_0 \cup \{\mu\})$ ; in this case, we pick some  $q_1 \in p^{-1}(\nu_1) \cap [q_0 + \mathcal{K}(X_0)]$ , which can be done by the acute angles assumption, and use the above construction again, starting with  $q_1$ , and obtaining a new set  $X_1 \subset X_0$  and a new segment  $\ell_1$ ; and iterate.

The above process eventually ends, because with each iteration, the size of the active set decreases. This construction yields monotonicity of  $X_i$  and the minimality property.

The above construction yields a specific sequence of  $q_i \in p^{-1}(\nu_i) \cap [q_{i-1} + \mathcal{K}(X_{i-1})]$ . We show in Appendix H of the full version that actually  $q_i \in p^{-1}(\nu_i) \cap (q_0 + X_{i-1}^\perp)$  and that the construction of  $L$  is independent of the choice of  $q_1, q_2, \dots, q_{k-1}$ .

**Part 2: Budget additivity for points on  $L$ .** Let  $\nu, \nu' \in$

$\text{im } L$  such that  $\nu \preceq \nu'$ . Let  $q \in \hat{Q}(\nu; q_0)$  and  $q' \in \hat{Q}(\nu'; q)$  such that  $q \in \hat{Q}(B; q_0)$  and  $q' \in \hat{Q}(B'; q)$ . In this part we show that  $q' \in \hat{Q}(B + B'; q_0)$ .

First, consider the case that  $\nu' = \mu$ . To see that  $q' \in \hat{Q}(B + B'; q_0)$ , first note that the constraints of Convex Program (2.1) hold, because  $U(q', \omega; q_0) = U(q', \omega; q) + U(q, \omega; q_0) \geq -B' - B$  for all  $\omega$  by path independence of the utility function. As noted in the introduction, in the absence of constraints, the utility  $U(\bar{q}, \mu; q_0)$  is maximized at any  $\bar{q}$  with  $p(\bar{q}) = \mu$ . Thus,  $q'$  is a global maximizer of the utility and satisfies the constraints, so  $q' \in \hat{Q}(B + B'; q_0)$ . If  $\nu = \mu$ , we must also have  $\nu' = \mu$  and the statement holds by previous reasoning.

In the remainder, we only analyze the case  $\nu \preceq \nu' \prec \mu$ . This means that  $\nu \in (\text{im } \ell_i) \setminus \{\nu_{i+1}\}$  and  $\nu' \in (\text{im } \ell_j) \setminus \{\nu_{j+1}\}$  for  $i \leq j$ . By Theorem 3.4, we therefore must have  $q \in [q_0 + \mathcal{K}(X_i)]$  and  $q' \in [q + \mathcal{K}(X_j)]$ . By anti-monotonicity of witness cones,  $\mathcal{K}(X_j) \supseteq \mathcal{K}(X_i)$  and hence,  $q' \in [q_0 + \mathcal{K}(X_j)]$ , yielding  $q' \in \hat{Q}(\nu'; q_0)$ .

We now argue that the budgets add up. Let  $x \in X_j \subseteq X_i$ . By Lemma 3.1, we obtain that  $q \in \hat{Q}(B; q_0)$  for  $B = -U(q, x; q_0)$ , and  $q' \in \hat{Q}(B'; q)$  for  $B' = -U(q', x; q)$ , and finally  $q' \in \hat{Q}(\bar{B}; q_0)$  for  $\bar{B} = -U(q', x; q_0)$ . However, by path independence of the utility function

$$\bar{B} = -U(q', x; q_0) = -U(q', x; q) - U(q, x; q_0) = B' + B.$$

**Part 3:  $L$  as the locus of all solutions.** See Appendix H of the full version for the proof that

$$\hat{Q}(q_0) = \bigcup_{\nu \in \text{im } L} \hat{Q}(\nu; q_0) .$$

**Part 3':  $L$  as the locus of solutions starting at a midpoint.** Let  $\nu \in \text{im } L$  and  $q \in \hat{Q}(\nu; q_0)$ . Since  $\hat{Q}(\nu; q_0) \subseteq p^{-1}(\nu) \cap (q_0 + X_\nu^\perp)$ , Part 1' (Appendix H of the full version) yields that the solution path  $L'$  for  $q$  coincides with the portion of  $L$  starting at  $\nu$ . Applying the proof of Part 3 to  $L'$ , we obtain

$$\hat{Q}(q) = \bigcup_{\nu' \in \text{im } L: \nu' \succeq \nu} \hat{Q}(\nu'; q) .$$

**Part 4: Proof of the theorem.** Let  $B, B' \geq 0$  and  $q \in \hat{Q}(B; q_0)$  and  $q' \in \hat{Q}(B'; q)$ . From Parts 3 and 3', we know that  $q \in \hat{Q}(\nu; q_0)$  and  $q' \in \hat{Q}(\nu'; q)$  for some  $\nu, \nu' \in \text{im } L$  such that  $\nu \preceq \nu'$ . By Part 2, we therefore obtain that  $q' \in \hat{Q}(B + B'; q_0)$ , proving the theorem.



## References

- [1] Jacob Abernethy, Yiling Chen, and Jennifer Wortman Vaughan. An optimization-based framework for automated market-making. In *ACM Conference on Electronic Commerce*, pages 297–306, 2011.
- [2] Alina Beygelzimer, John Langford, and David M. Pennock. Learning performance of prediction markets with Kelly bettors. In *International Conference on Autonomous Agents and Multiagent Systems*, pages 1317–1318, 2012.
- [3] GW Brier. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78:13, 1950.
- [4] Yiling Chen and David M. Pennock. A utility framework for bounded-loss market makers. In *Conference on Uncertainty in Artificial Intelligence*, pages 49–56, 2007.
- [5] Yiling Chen and Jennifer Wortman Vaughan. A new understanding of prediction markets via no-regret learning. In *ACM Conference on Electronic Commerce*, pages 189–198, 2010.
- [6] Miroslav Dudík, Sébastien Lahaie, and David M. Pennock. A tractable combinatorial market maker using constraint generation. In *ACM Conference on Electronic Commerce*, 2012.
- [7] Edmund Eisenberg and David Gale. Consensus of subjective probabilities: The pari-mutuel method. *Annals of Mathematical Statistics*, 30:165–168, 1959.
- [8] Lance Fortnow and Rahul Sami. Multi-outcome and multidimensional market scoring rules. *CoRR*, abs/1202.1712, 2012.
- [9] Rafael Frongillo, Nicolas Della Penna, and Mark Reid. Interpreting prediction markets: a stochastic approach. In *Advances in Neural Information Processing Systems 25*, pages 3275–3283. 2012.
- [10] Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477): 359–378, 2007.
- [11] Sharad Goel, Daniel M. Reeves, Duncan J. Watts, and David M. Pennock. Prediction without markets. In *ACM Conference on Electronic Commerce*, pages 357–366, New York, NY, USA, 2010. ACM.
- [12] Robin Hanson. Combinatorial information market design. *Information Systems Frontiers*, 5(1):107–119, 2003.
- [13] Robin Hanson. Logarithmic market scoring rules for modular combinatorial information aggregation. *Journal of Prediction Markets*, 1(1):3–15, February 2007.
- [14] Robert A. Jacobs. Methods for combining experts’ probability assessments. *Neural Computation*, 7(5): 867–888, September 1995.
- [15] Xiaolong Li and Jennifer Wortman Vaughan. An axiomatic characterization of adaptive-liquidity market makers. In *ACM Conference on Electronic Commerce*, 2013.
- [16] Charles F. Manski. Interpreting the predictions of prediction markets. *Economics Letters*, 91(3):425–429, June 2006.
- [17] Abraham Othman, Tuomas Sandholm, David M. Pennock, and Daniel M. Reeves. A practical liquidity-sensitive automated market maker. In *ACM Conference on Electronic Commerce*, 2010.
- [18] R. Tyrrell Rockafellar. *Convex Analysis*. Princeton University Press, Princeton, New Jersey, 1970.
- [19] Justin Wolfers and Eric Zitzewitz. Prediction markets. *Journal of Economic Perspectives*, Winter, 2004.
- [20] Justin Wolfers and Eric Zitzewitz. Interpreting prediction market prices as probabilities. IZA Discussion Papers 2092, Institute for the Study of Labor (IZA), April 2006.

---

# A Probabilistic Logic for Reasoning about Uncertain Temporal Information

---

**Dragan Doder**

Computer Science and Communication  
University of Luxembourg  
6, rue Coudenhove-Kalergi L-1359 Luxembourg  
dragan.doder@uni.lu

**Zoran Ognjanović**

Mathematical Institute  
Serbian Academy of Sciences and Arts  
Kneza Mihaila 36, 11000 Belgrade, Serbia  
zorano@mi.sanu.ac.rs

## Abstract

The main goal of this work is to present the proof-theoretical and model-theoretical approach to a probabilistic logic which allows reasoning about temporal information. We extend both the language of linear time logic and the language of probabilistic logic, allowing statements like “A will always hold” and “the probability that A will hold in next moment is at least the probability that B will always hold,” where A and B are arbitrary statements. We axiomatize this logic, provide corresponding semantics and prove that the axiomatization is sound and strongly complete. We show that the problem of deciding decidability is PSPACE-complete, no worse than that of linear time logic.

## 1 INTRODUCTION

The study of temporal logics started with the seminal work of Arthur Prior [Prior, 1957]. Temporal logics are designed in order to analyze and reason about the way that systems change over time, and have been shown to be a useful tool in describing behavior of an agent’s knowledge base, for specification and verification of programs, hardware, protocols in distributed systems etc. [Emerson, 1990, Emerson, 1995]. In many practical situations the temporal information is not known with certainty. A typical example is formal representation of information about tracking moving objects with GPS systems, in the case in which the locations or the identities of the objects are not certainly known [Grant et al., 2010].

Many different tools are developed for representing, and reasoning with, uncertain knowledge. One particular line of research concerns the formalization in terms of probabilistic logic. After Nilsson [Nilsson, 1986] gave a procedure for probabilistic entailment which, given probabilities of premises, calculates bounds on the probabilities of the

derived sentences, researchers from the field started investigation about formal systems for probabilistic reasoning. [Fagin et al., 1990] provided a finitary axiomatization for reasoning about linear combinations of probabilities, and they proved weak completeness (every consistent formula is satisfiable). Their formulas are Boolean combinations of the expressions of the form  $r_1w(\alpha_1) + \dots + r_nw(\alpha_n) \geq r_{n+1}$ , where  $w$  is the probability operator and  $\alpha_i$ ’s are propositional formulas. The semantics of the logic use finitely additive probabilities, since  $\sigma$ -additivity cannot be expressed by a formula of their language.

In this paper, we extend the approach from [Fagin et al., 1990]. We start with the propositional linear time logic (LTL) [Gabbay et al., 1980] with the “next” operator  $\bigcirc$  and “until” operator  $U$ . The meaning of the formula  $\bigcirc\alpha$  is “ $\alpha$  holds in the next time instance”, and  $\alpha U\beta$  we read “ $\alpha$  holds in every time instance until  $\beta$  holds”. We apply the probabilistic operator  $w$  to the formulas of LTL and define probabilistic formulas using the linear combinations, like in [Fagin et al., 1990]. In our logic there are two types of formulas, LTL formulas and probabilistic formulas, with the requirement that if an LTL formula is true, then its probability is equal to 1.

The main technical challenge in axiomatizing such a logic lies in the fact that the set of models of the formula  $\alpha U\beta$  can be represented as a countable union of models of temporal formulas which are pairwise disjoint. As a consequence, finitely additive semantics is obviously not appropriate for such a logic, and we propose  $\sigma$ -additive semantics for the logic. On the other hand, expressing  $\sigma$ -additivity with an axiom would require infinite disjunctions, and the resulting logic would be undecidable. We shown in Section 3.1 that any finitary axiomatic system wouldn’t be complete for the  $\sigma$ -additive semantics.

In order to overcome this problem, we axiomatize our language using infinitary rules of inference. Thus, in this work the term “infinitary” concerns the meta language only, i.e., the object language is countable and the formulas are finite, while only proofs are allowed to be infinite. We prove that our axiomatization is sound and strongly complete (every

consistent set of formulas is satisfiable). We also prove that the logic is decidable, and we show that the satisfiability problem is *PSPACE*-complete, no harder than satisfiability for LTL.

There are several logics which combine time and probability in different ways [Guelev, 2000, Haddawy, 1996, Halpern and Pucella, 2006, Hansson and Jonsson, 1994, Ognjanovic, 2006, Shakarian et al., 2011]. However, to the best of our knowledge, this is the first complete axiomatization for the  $\sigma$ -additive probabilistic semantics.

## 2 THE LOGIC $PL_{LTL}$ : SYNTAX AND SEMANTICS

We present the syntax and semantics of the logic for probabilistic reasoning about linear time formulas, that we denote by  $PL_{LTL}$ . The logic contains two types of formulas: formulas of LTL without probabilities, and the linear weight formulas in the style of [Fagin et al., 1990], with weights applied to temporal formulas.

In order to give semantics to the formulas, we first briefly review some probability theory [Ash and Doléans-Dade, 1999]. If  $W \neq \emptyset$ , then  $H$  is an algebra of subsets of  $W$ , if it is a set of subsets of  $W$  such that:

- (a)  $W \in H$ ,
- (b) if  $A, B \in H$ , then  $W \setminus A \in H$  and  $A \cup B \in H$ .

A function  $\mu : H \rightarrow [0, 1]$  is a ( $\sigma$ -additive) probability measure, if the following conditions hold:

- (1)  $\mu(W) = 1$ ,
- (2)  $\mu(\bigcup_{i \in \omega} A_i) = \sum_{i \in \omega} \mu(A_i)$ , whenever  $A, A_i \in H$  and  $A_i \cap A_j = \emptyset$  for all  $i \neq j$ .

For  $W, H$  and  $\mu$  described above, the triple  $\langle W, H, \mu \rangle$  is called a probability space. A function  $\mu : H \rightarrow [0, 1]$  is a finitely additive probability measure, if the condition

- (3)  $\mu(A \cup B) = \mu(A) + \mu(B)$ , whenever  $A \cap B = \emptyset$ .

holds, instead of (2). We also say that an algebra  $H$  is a  $\sigma$ -algebra, if  $\bigcup_{i \in \omega} A_i \in H$  whenever  $A_i \in H$  for every  $i \in \omega$ .

For a finitely additive  $\mu$ , the condition (2) is equivalent to the condition

- (2')  $\mu(\bigcup_{i \in \omega} A_i) = \lim_{n \rightarrow +\infty} \mu(\bigcup_{i=0}^n A_i)$ .

We will actually use (2') in the axiomatization of our logic (see the inference rule R6).

### 2.1 SYNTAX

First we introduce LTL formulas. Suppose that  $\mathcal{P}$  is a nonempty finite set of propositional letters. We denote the

elements of  $\mathcal{P}$  with  $p$  and  $q$ , possibly with subscripts.

**Definition 1 (LTL formula)** An LTL formula is any formula built from propositional letters from  $\mathcal{P}$ , using the Boolean connectives  $\neg$  and  $\wedge$ , and the temporal operators  $\bigcirc$  and  $U$ .

We use  $For_{LTL}$  for the set of all state formulas and denote arbitrary LTL formulas by  $\alpha, \beta$  and  $\gamma$ , possibly with subscripts.

We use  $\neg$  and  $\wedge$  as the primitive connectives, while other Boolean connectives ( $\rightarrow, \vee, \leftrightarrow$ ) can be introduced as usual. We also define other LTL operators  $F$  (sometime) and  $G$  (always) as abbreviations:  $F\alpha$  is  $\bigvee U\alpha$ , and  $G\alpha$  is  $\neg F\neg\alpha$ . Note that we use the strong version of  $U$ , which means that if  $\alpha U \beta$  holds in a path, then  $\beta$  must hold eventually.

**Example 1** The expression

$$\bigcirc(p \wedge q) \rightarrow (pU\neg q)$$

is an example of LTL formula. Its meaning is “if both  $p$  and  $q$  hold in the next moment, then  $p$  will hold until  $q$  becomes false”.

Semantics for LTL formulas consists of the set of paths, where a path is a  $\omega$ -structure in  $\mathcal{P}$ , of the form  $\sigma = s_0, s_1, s_2, \dots$ . Here  $s_i$ , called the  $i$ -th time instance of  $\sigma$ , is a subset of  $\mathcal{P}$ , and  $p \in s_i$  represent the propositional letter  $p$  being true at time  $i$  in  $\sigma$ . We denote the set of all paths with  $\bar{\Sigma}$ . In the rest of the paper, we use the following abbreviations:

- $\sigma_{\geq i}$  is the path  $s_i, s_{i+1}, s_{i+2}, \dots$
- $\sigma_i$  is the state  $s_i$ .

The evaluation function<sup>1</sup>  $v : \bar{\Sigma} \times For_{LTL} \rightarrow \{0, 1\}$  is defined recursively as follows:

- if  $p \in \mathcal{P}$ , then  $v(\sigma, p) = 1$  iff  $p \in \sigma_0$ ,
- $v(\sigma, \neg\alpha) = 1$  iff  $v(\sigma, \alpha) = 0$ ,
- $v(\sigma, \alpha \wedge \beta) = 1$  iff  $v(\sigma, \alpha) = 1$  and  $v(\sigma, \beta) = 1$ ,
- $v(\sigma, \bigcirc\alpha) = 1$  iff  $v(\sigma_{\geq 1}, \alpha) = 1$ ,
- $v(\sigma, \alpha U \beta) = 1$  iff there is some  $i \in \omega$  such that  $v(\sigma_{\geq i}, \beta) = 1$ , and for each  $j \in \omega$ , if  $0 \leq j < i$  then  $v(\sigma_{\geq j}, \alpha) = 1$ .

<sup>1</sup>In the literature, the evaluation of LTL formulas in paths is usually given in terms of satisfiability relation  $\models$ . We do not follow this notation, because in this paper we use  $\models$  to denote satisfiability of formulas in  $PL_{LTL}$ -structures.

We say that  $\alpha$  is true in the path  $\sigma$ , if  $v(\sigma, \alpha) = 1$ .

Now we introduce the probabilistic formulas. By  $\mathcal{Q}$  we denote the set of rational numbers. First we define the probabilistic terms.

**Definition 2 (Probabilistic term)** A probabilistic term is any expression of the form

$$r_1 w(\alpha_1) + \dots + r_k w(\alpha_k) + r_{k+1},$$

where  $k$  is a positive integer, and for all  $i \leq k + 1$ ,  $\alpha_i \in For_{LTL}$  and  $r_i \in \mathcal{Q}$ .<sup>2</sup>

We use  $f$  and  $g$ , possibly subscripted, to denote probabilistic terms.

**Definition 3 (Probabilistic formula)** A basic probabilistic formula is any formula of the form  $f \geq r$ , where  $f$  is a probabilistic term and  $r \in \mathcal{Q}$ . The set  $For_P$  of probabilistic formulas is the smallest set containing all basic probabilistic formulas, closed under Boolean connectives.

We denote by  $\phi, \psi$  and  $\theta$  (possibly with indices) the elements of  $For_P$ . To simplify notation, we define the following abbreviations:  $f \geq g$  is  $f - g \geq 0$ ,  $f \leq g$  is  $g \geq f$ ,  $f < g$  is  $\neg f \geq g$ ,  $f > g$  is  $\neg f \leq g$  and  $f = g$  is  $f \geq g \wedge f \leq g$ .

**Example 2** The expression

$$w(p \vee q) = w(\bigcirc p) \rightarrow w(Gq) \leq \frac{1}{2}$$

is a probabilistic formula. Its meaning is “if the probability that either  $p$  or  $q$  hold in this moment is equal to the probability that  $p$  will hold in the next moment, then the probability that  $q$  will always hold is at most one half”.

**Definition 4 (Formula)** The set  $For$  of all formulas of the logic  $PL_{LTL}$  is  $For = For_{LTL} \cup For_P$ .

We denote arbitrary formulas by  $\Phi$  and  $\Psi$  (possibly with subscripts). We denote by  $\perp$  both  $\phi \wedge \neg\phi$  and  $\alpha \wedge \neg\alpha$ , letting the context determines the meaning. Similarly, we use  $\top$  for both LTL and probabilistic formulas.

**Example 3** The expression

$$(p \vee \bigcirc q) \rightarrow w(p \vee \bigcirc q) = 1$$

is not a formula, since mixing LTL formulas and probabilistic formulas is not allowed, by Definition 4.

<sup>2</sup>In [Fagin et al., 1990],  $r_{k+1}$  does not appear in the definition of terms. We introduce it for the simpler presentation, when we introduce other formulas as abbreviations.

## 2.2 SEMANTICS

The semantics of the logic  $PL_{LTL}$  is based on the possible-world approach.

**Definition 5 ( $PL_{LTL}$  structure)** A  $PL_{LTL}$  structure is a tuple  $M = \langle W, H, \mu, \pi \rangle$  where:

- $W$  is a nonempty set of worlds,
- $\langle W, H, \mu \rangle$  is a probability space, and
- $\pi : W \rightarrow \bar{\Sigma}$  provides for each world  $w \in W$  a path  $\pi(w)$ .

For a  $PL_{LTL}$  structure  $M = \langle W, H, \mu, \pi \rangle$ , we define

$$[\alpha]_M = \{w \in W \mid v(\pi(w), \alpha) = 1\}.$$

We say that  $M$  is measurable, if  $[\alpha]_M \in H$  for every  $\alpha \in For_{LTL}$ . We denote the class of all measurable  $PL_{LTL}$  structures with  $PL_{LTL}^{Meas}$ .

Now we define the satisfiability of a formula from  $For$  in a structure from  $PL_{LTL}^{Meas}$ .

**Definition 6 (Satisfiability)** Let  $M = \langle W, H, \mu, \pi \rangle$  be a  $PL_{LTL}$  structure. We define the satisfiability relation  $\models_{\subseteq} PL_{LTL}^{Meas} \times For$  recursively as follows:

- $M \models \alpha$  iff  $v(\pi(w), \alpha) = 1$  for every  $w \in W$ ,
- $M \models r_1 w(\alpha_1) + \dots + r_k w(\alpha_k) \geq r$  iff  $r_1 \mu([\alpha_1]_M) + \dots + r_k \mu([\alpha_k]_M) \geq r$ ,
- $M \models \neg\phi$  iff  $M \not\models \phi$ ,
- $M \models \phi \wedge \psi$  iff  $M \models \phi$  and  $M \models \psi$ .

**Definition 7 (Model)** We say that  $M \in PL_{LTL}^{Meas}$  is a model of  $\Phi$ , if  $M \models \Phi$ . A formula  $\Phi$  is valid, if  $M \models \Phi$  holds for every  $M \in PL_{LTL}^{Meas}$ . We say that  $M$  is a model of a set of formulas  $T$ , and we write  $M \models T$ , iff  $M \models \Phi$  for every  $\Phi \in T$ . A set of formulas  $T$  is satisfiable if there is  $M$  such that  $M \models T$ .

**Definition 8 (Entailment)** We say that the set of formulas  $T$  entails a formula  $\Phi$ , and we write  $T \models \Phi$ , if all  $M \in PL_{LTL}^{Meas}$ ,  $M \models T$  implies  $M \models \Phi$ .

For every  $\alpha, \beta \in For_{LTL}$ , let us denote by  $\alpha \bar{U}_n \beta$  the formula

$$\bigwedge_{k=0}^{n-1} \bigcirc^k \alpha \wedge \bigcirc^n \beta,$$

and by  $\alpha U_n \beta$  the formula  $\bigvee_{k=0}^n \alpha \bar{U}_k \beta$ .

Those formulas will play the important role in our axiomatization. Obviously,  $v(\sigma, \alpha U \beta) = 1$  iff there is some  $n \in \omega$  such that  $v(\sigma, \alpha \bar{U}_n \beta) = 1$ , and

$$[\alpha U \beta]_M = \bigcup_{n \in \omega} [\alpha \bar{U}_n \beta]_M. \quad (1)$$

Similarly,

$$[\alpha U \beta]_M = \bigcup_{n \in \omega} [\alpha U_n \beta]_M. \quad (2)$$

Since (1) follows directly from the definition of the evaluation function  $v$ , we will use it to properly axiomatize LTL part of our logic. On the other hand, (2) is more convenient for capturing  $\sigma$ -additivity.

### 3 The axiomatization of $PL_{LTL}$

In this section we provide an axiomatization for  $PL_{LTL}$ , which we denote by  $AX_{PL_{LTL}}$ . Let us first discuss some axiomatization issues. By (2) and  $\sigma$ -additivity, we obtain  $\mu([\alpha U \beta]_M) = \mu(\bigcup_{n \in \omega} [\alpha U_n \beta]_M)$ . Then we can see that the set

$$T = \{w(\alpha U \beta) > r\} \cup \{w(\alpha U_n \beta) \leq r \mid n \in \omega\}$$

is an unsatisfiable set of formulas. On the other hand, it is easy to check that every finite subset of  $T$  is satisfiable. In other words, the logic is not compact. It is known that, in this case, any finitary axiomatization would be incomplete [van der Hoek, 1997]. Here we use an infinitary rule (R6) to obtain completeness, and, in particular, to make the set  $T$  inconsistent. It turns that it is necessary (see the proof of Theorem 4) to introduce another infinitary rule (R4) to properly axiomatize LTL part of the logic, since the set of LTL formulas  $\{\alpha U \beta\} \cap \{\neg(\alpha \bar{U}_n \beta) \mid n \in \omega\}$  is also an example of non-compactness.

#### 3.1 THE AXIOMATIC SYSTEM $AX_{PL_{LTL}}$

the axiomatization  $AX_{PL_{LTL}}$  contains 8 axioms and 6 rules of inference. We divide the axioms into 3 groups as given below.

Tautologies

A1. All instances of classical propositional tautologies for both LTL and probabilistic formulas.

Temporal axioms

A2.  $\bigcirc(\alpha \rightarrow \beta) \rightarrow (\bigcirc\alpha \rightarrow \bigcirc\beta)$ .

A3.  $\neg \bigcirc \alpha \leftrightarrow \bigcirc \neg \alpha$ .

A4.  $\alpha U \beta \leftrightarrow \beta \vee (\alpha \wedge \bigcirc(\alpha U \beta))$ .

Axioms for reasoning about linear inequalities

A5. All instances of valid formulas about linear inequalities.

Probabilistic axioms

A6.  $w(\alpha) \geq 0$ .

A7.  $w(\alpha \wedge \beta) + w(\alpha \wedge \neg \beta) = w(\alpha)$ .

A8.  $w(\alpha \rightarrow \beta) = 1 \rightarrow w(\alpha) \leq w(\beta)$ .

Inference rules

R1. From  $\Phi$  and  $\Phi \rightarrow \Psi$  infer  $\Psi$  (where either  $\Phi, \Psi \in For_{LTL}$  or  $\Phi, \Psi \in For_P$ ).

R2. From  $\alpha$  infer  $\bigcirc\alpha$ .

R3. From  $\alpha$  infer  $w(\alpha) = 1$ .

R4. From the set of premises

$$\{\gamma \rightarrow \neg(\alpha \bar{U}_n \beta) \mid n \in \omega\}$$

infer  $\gamma \rightarrow \neg(\alpha U \beta)$ .

R5. From the set of premises

$$\{\phi \rightarrow f \geq r - \frac{1}{n} \mid n \in \omega \setminus \{0\}\}$$

infer  $\phi \rightarrow f \geq r$ .

R6. From the set of premises

$$\{\phi \rightarrow w(\alpha U_n \beta) \leq r \mid n \in \omega\}$$

infer  $\phi \rightarrow w(\alpha U \beta) \leq r$ .

Let us briefly discuss the axiomatic system.

A1 and R1 allow propositional reasoning with all formulas from  $For$ .

The axioms A2–A4 are some standard axioms in various axiomatization of LTL. Although all the axiomatizations contain some additional axioms, we show in Lemma 1(1) that all the valid temporal formulas can be deduced in  $AX_{PL_{LTL}}$ . Moreover, by Lemma 2, A1–A4 together with R1, R2 and R4 make a strongly complete system for LTL. Note that we use the temporal necessitation R2 with the next operator, while the standard generalization can be derived, as it is shown in the proof of Lemma 1(1). The rule R4 is an infinitary rule that characterizes the until operator. It is similar to a rule from [Marinkovic et al., 2014], and it is necessary for the proof of  $\sigma$ -additivity.

The axiom A5 includes all valid formulas about linear inequalities. For example,  $f + 1 \leq f + 2$  and  $f + g =$

$g + f$  are instances of A5. A particular sound and complete axiomatization for Boolean combination is given in [Fagin et al., 1990], but, as it is pointed out there, any other axiomatization can be used.

The probabilistic axioms A6 and A7 correspond to non-negativity and finite additivity, respectively. They are two of the four axioms presented in [Fagin et al., 1990]. Other two axioms are theorems of  $AX_{PLTL}$  (see Lemma 1). The rule R3 states that if we know that  $\alpha$  holds, then we believe that it is true with probability 1. The rules R4–R6 are infinitary rules of inference. R4 and R6 are crucial for the proof of  $\sigma$ -additivity, while R5, ensures that the values of probability measures belong to the set of reals. R5 is a variant of a rule introduced in [Perovic et al., 2008].

**Definition 9 (Proof)** A formula  $\Phi$  is a theorem of the logic  $PLTL$ , ( $\vdash \Phi$ ), if there is an at most countable sequence of formulas  $\Phi_0, \Phi_1, \dots, \Phi$ , such that every  $\Phi_i$  is an axiom, or it is derived from the preceding formulas by an inference rule.

A formula  $\Phi$  is deducible from a set of formulas  $T$  ( $T \vdash \Phi$ ) if there is an at most countable sequence of formulas  $\Phi_0, \Phi_1, \dots, \Phi$ , such that every  $\Phi_i$  is a theorem or a formula from  $T$ , or it is derived from the preceding formulas by one of the inference rules, excluding R2. The corresponding sequence  $\Phi_0, \Phi_1, \dots, \Phi$  is the proof of  $\Phi$  from  $T$ .

By the previous definition, application of the rule R2 is restricted to theorems only. Otherwise, any change during the time would be impossible. Note that the length of a proof (the number of formulas in the corresponding sequence) is any countable successor ordinal.

**Definition 10 (Consistency)** A set of formulas  $T$  is consistent if there is no  $\phi \in For_P$  such that  $T \vdash \phi \wedge \neg\phi$ , otherwise it is inconsistent.  $T$  is maximal consistent if it is consistent and for all  $\Phi \notin T$ ,  $T \cup \{\Phi\}$  is inconsistent.

Next we make several observations about the notions of consistency and maximal consistency:

- If  $T$  is consistent, then there is no  $\alpha \in For_{LTL}$  such that  $T \vdash \alpha \wedge \neg\alpha$ , since otherwise  $T \vdash w(\alpha) = 1 \wedge w(\neg\alpha) = 1$  by R3, and  $T \vdash w(\alpha) = 1 \wedge \neg w(\alpha) = 1$  by probabilistic axioms.

- Maximal consistency of  $T$  doesn't imply that for every  $\alpha \in For_{LTL}$  either  $T \vdash \alpha$  or  $T \vdash \neg\alpha$ . Indeed, suppose that  $w(\alpha) = \frac{1}{2} \in T$  for some  $\alpha$ . If  $T \vdash \alpha$  or  $T \vdash \neg\alpha$ , then by R3 (and some probabilistic reasoning) we have  $T \vdash w(\alpha) = 1$  or  $T \vdash w(\alpha) = 0$ , which would make  $T$  inconsistent. On the other hand, for a  $\phi \in For_P$  we have either  $T \vdash \phi$  or  $T \vdash \neg\phi$  (see Lemma 1(4)).

- If  $T$  is consistent, then  $T$  is *deductively closed*, i.e., if  $T \vdash \Phi$  then  $\Phi \in T$ .

### 3.2 SOME THEOREMS ABOUT $AX_{PLTL}$

It is straightforward to check that all the axioms of  $AX_{PLTL}$  are valid, and that the rules of inference maintain the validity of formulas. Thus, we omit the proof of the following result.

**Theorem 1 (Soundness)** The axiomatization  $AX_{PLTL}$  is sound with respect to the class of models  $PL_{LTL}^{Meas}$ .

**Theorem 2 (Deduction theorem)** Let  $T$  be a set of formulas and let  $\Phi$  and  $\Psi$  be two formulas such that either  $\Phi, \Psi \in For_{LTL}$  or  $\Phi, \Psi \in For_{LTL}$ . Then  $T \cup \{\Phi\} \vdash \Psi$  iff  $T \vdash \Phi \rightarrow \Psi$ .

*Proof. (sketch)* We will prove the direction from right to left because the other direction is immediate from R1. We will use induction on the length of the inference. We will only consider the case when R6 is applied. Suppose that  $T \cup \{\phi\} \vdash \psi \rightarrow w(\alpha U \beta) \leq r$  is obtained by R6. Then  $T \cup \{\phi\} \vdash \psi \rightarrow w(\alpha U_n \beta) \leq r$  holds, by assumption, for every  $n \in \omega$ . Using induction hypothesis and reasoning as above, we have:

$T \vdash \phi \rightarrow (\psi \rightarrow w(\alpha U_n \beta) \leq r)$ , for every  $n \in \omega$ ;

$T \vdash (\phi \wedge \psi) \rightarrow w(\alpha U_n \beta) \leq r$ , for every  $n \in \omega$ ;

$T \vdash (\phi \wedge \psi) \rightarrow w(\alpha U \beta) \leq r$ , by R6;

$T \vdash \phi \rightarrow (\psi \rightarrow w(\alpha U \beta) \leq r)$ .

#### Lemma 1

1. If  $v(\sigma, \alpha) = 1$  for all  $\sigma \in \bar{\Sigma}$ , then  $\vdash \alpha$ .
2.  $\vdash w(\top) = 1$
3. If  $T \vdash \alpha \leftrightarrow \beta$ , then  $T \vdash w(\alpha) = w(\beta)$
4. If  $T$  is maximal consistent then either  $\phi \in T$  or  $\neg\phi \in T$ , for every  $\phi \in For_P$ .

*Proof.* (1) It is sufficient to prove that all the axioms of any complete axiomatization of LTL (for example C1–C8 form [Reynolds, 2001]) are theorems of our logic, and that the standard Generalization rule “if  $\alpha$  is a theorem, from  $\alpha$  infer  $G\alpha$ ” is derived rule in  $AX_{PLTL}$ . As an illustration, let us derive Generalization. If  $\vdash \alpha$ , applying rule R2 we obtain  $\vdash \bigcirc^n \alpha$  for every  $n \in \omega$ . Using A3, we conclude  $\vdash \neg \bigcirc^n \neg \alpha$  for every  $n \in \omega$ . Note that  $\neg \bigcirc^n \neg \alpha$  can be written as  $\neg(\top U_n \neg \alpha)$ . Finally, applying R4 we obtain  $\vdash \neg(\top U \neg \alpha)$ , or, equivalently,  $\vdash G\alpha$ .

(2) Follows directly from R3.

(3) Apply R3, then A8.

(4) If  $\phi \notin T$ , then  $T \cup \{\phi\} \vdash \perp$ , by the maximality of  $T$ . By Theorem 2, we have  $T \vdash \phi \rightarrow \perp$ , so  $T \vdash \neg\phi$ . Similarly, if  $\phi \in T$ , then  $T \vdash \phi$ , which contradicts the assumption that  $T$  is consistent.

Let us comment the lemma. By (1), we can use all the standard theorems of LTL in our reasoning in  $PL_{LTL}$ . (2) is an

axiom for probabilistic reasoning from [Fagin et al., 1990]. (3) plays the crucial role in the construction of the canonical model in the next section. If we choose  $\alpha$  and  $\beta$  to be propositional formulas and  $T = \emptyset$ , we obtain another axiom from [Fagin et al., 1990]. Thus, by (1)–(3),  $AX_{PLTL}$  extends both temporal and probabilistic logic.

We use (4) in the proof of Theorem 5. We already pointed out that the same property doesn't hold for the LTL formulas. Note that we cannot copy the proof of (4) in LTL case, since we distinguish between the probabilistic contradiction and LTL contradiction (although we use  $\perp$  in both cases).

## 4 THE COMPLETENESS OF $PL_{LTL}$

In this section we prove strong version of completeness theorem: “every consistent set of formulas has a model”. We use a Henkin-like construction. First we extend a consistent set  $T$  of formulas to a maximal consistent set  $T^*$ , then we use  $T^*$  to define the corresponding structure  $M_{T^*}$ , and finally we prove that  $M_{T^*}$  is a model of  $T$ . For given  $T^*$ , we say that  $M_{T^*}$  is its canonical model.

### 4.1 LINDENBAUM'S LEMMA

**Theorem 3 (Lindenbaum's lemma)** *Every consistent set of formulas can be extended to a maximal consistent set.*

*Proof.(sketch)* Let  $T$  be a consistent set and let  $\Phi_0, \Phi_1, \dots$  be an enumeration of all formulas from  $For$ . We define the sequence of sets  $T_i$ ,  $i = 0, 1, 2, \dots$  and the set  $T^*$  recursively as follows:

1.  $T_0 = T$ ,
2. for every  $i \geq 0$ ,
  - (a) if  $T_i \cup \{\Phi_i\}$  is consistent, then  $T_{i+1} = T_i \cup \{\Phi_i\}$ , otherwise
  - (b) if  $\Phi_i$  is of the form  $\gamma \rightarrow \neg(\alpha U \beta)$ , then  $T_{i+1} = T_i \cup \{\gamma \rightarrow (\alpha \bar{U}_n \beta)\}$ , where  $n$  is the smallest nonnegative integer such that  $T_{i+1}$  is consistent, otherwise
  - (c) if  $\Phi_i$  is of the form  $\phi \rightarrow f \geq r$ , then  $T_{i+1} = T_i \cup \{\phi \rightarrow f < r - \frac{1}{n}\}$ , where  $n$  is the smallest positive integer such that  $T_{i+1}$  is consistent, otherwise
  - (d) if  $\Phi_i$  is of the form  $\phi \rightarrow w(\alpha U \beta) \leq r$ , then  $T_{i+1} = T_i \cup \{\phi \rightarrow w(\alpha U_n \beta) > r\}$ , where  $n$  is the smallest nonnegative integer such that  $T_{i+1}$  is consistent, otherwise
  - (e)  $T_{i+1} = T_i$ .
3.  $T^* = \bigcup_{i=0}^{\infty} T_i$ .

First, using Theorem 2 one can prove that the set  $T^*$  is correctly defined, i.e., there exist  $n$  from the parts 2(b)–2(d) of the construction. Each  $T_i$ ,  $i > 0$  is consistent. The steps (1) and (2) of the construction ensure that  $T^*$  is maximal. Also,  $T^*$  obviously doesn't contain all formulas. Finally, one can show that  $T^*$  is deductively closed set, and as a consequence we obtain that  $T^*$  is consistent (otherwise it would contain  $\perp$ ).

### 4.2 CANONICAL MODEL

**Definition 11 (Canonical model)** *For a maximal consistent set  $T^*$ , we define a  $PL_{LTL}$  structure as a tuple  $M_{T^*} = \langle W, H, \mu, \pi \rangle$ , such that:*

1.  $W = \{\sigma \in \bar{\Sigma} \mid v(\sigma, \alpha) = 1 \text{ for all } \alpha \in T^* \cap For_{LTL}\}$ ,
2.  $H = \{[\alpha] \mid \alpha \in For_{LTL}\}$ , where  $[\alpha] = \{w \in W \mid v(w, \alpha) = 1\}$ ,
3.  $\mu([\alpha]) = \sup\{r \in \mathcal{Q} \mid T^* \vdash w(\alpha) \geq r\}$ , for every  $\alpha \in For_{LTL}$ ,
4.  $\pi(w) = w$  for every  $w \in W$ .

Now we show that  $M_{T^*}$  is a measurable  $PL_{LTL}$  structure. In the proof, we will use the following result.

**Lemma 2** *The axioms A1–A4 and the inference rules R1, R2 and R4 form a strongly complete axiomatization for LTL.*

*Proof.* We need to show that every consistent set  $T$  of LTL formulas has a model, i.e., that there is  $\sigma$  such that  $v(\sigma, \alpha) = 1$  for every  $\alpha \in T$ . Reasoning similarly as above, we can prove that Deduction theorem holds and that  $T$  can be extended to a maximal consistent set  $T^*$ . Now we work with LTL formulas only, and we can prove that for each  $\alpha$  either  $\alpha \in T^*$  or  $\neg\alpha \in T^*$ . Also, using the axiomatization it is straightforward to show that if  $T^*$  is maximal consistent set, then the set  $T_n^* = \{\alpha \mid \bigcirc \alpha \in T^*\}$  is also maximal consistent.

For given  $T^*$ , we define the path  $\sigma = s_0, s_1, \dots$  by  $s_i = \{p \in \mathcal{P} \mid T_i^* \vdash p\}$ .

It is sufficient to prove that  $v(\sigma, \gamma) = 1$  iff  $T^* \vdash \gamma$ , for every LTL formula  $\gamma$ . We use induction on the complexity of the formula. The only interesting case is when  $\gamma$  is of the form  $\alpha U \beta$ .

$v(\sigma, \gamma) = 0$  iff  $v(\sigma, \neg(\alpha U \beta)) = 1$

iff for all  $n \in \omega$ , it is not the case that  $v(\sigma_{\geq n}, \beta) = 1$  and

for all  $k < n$ ,  $v(\sigma_{\geq k}, \alpha) = 1$

iff for all  $n \in \omega$ , it is not the case that  $T_n^* \vdash \beta$  and for all  $k < n$ ,  $T_k^* \vdash \alpha$  (by induction hypothesis)

iff for all  $n \in \omega$ , it is not the case that  $T^* \vdash \bigcirc^n \beta$  and for all  $k < n$ ,  $T^* \vdash \bigcirc^k \alpha$

iff for all  $n \in \omega$ ,  $T^* \vdash \neg(\alpha \bar{U}_n \beta)$  (by the maximal consistency of  $T^*$ )  
iff  $T^* \vdash \neg(\alpha U \beta)$  (by R4).

**Theorem 4** For every maximal consistent set  $T^*$ ,  $M_{T^*} \in PL_{LTL}^{Meas}$ .

*Proof.* First we need to show that the definition is correct. The set  $\{[\alpha] \mid \alpha \in For_{LTL}\}$  is an algebra of subsets of  $W$ , since  $W = [\top]$ ,  $W \setminus [\alpha] = [\neg\alpha]$  and  $[\alpha] \cup [\beta] = [\alpha \vee \beta]$ . We also need to check that  $\mu$  is correctly defined, i.e., that if  $[\alpha] = [\beta]$  then  $\mu([\alpha]) = \mu([\beta])$ . From  $[\alpha] = [\beta]$  we conclude that if  $\sigma$  is a path such that  $v(\sigma, \gamma) = 1$  for all  $\gamma \in T^* \cap For_{LTL}$ , then  $v(\sigma, \alpha \leftrightarrow \beta) = 1$ . From Lemma 2 we obtain  $T^* \vdash \alpha \leftrightarrow \beta$ . Consequently,  $T^* \vdash w(\alpha) = w(\beta)$  by Lemma 1(3), so  $\mu([\alpha]) = \mu([\beta])$ . Obviously  $\mu(W) = \mu([\top]) = 1$  by Lemma 1(2). Similarly, using A6 we conclude that  $\mu$  is nonnegative, and using A7 we conclude that  $\mu$  is a finitely additive probability measure on  $A$ . We need to prove that  $\mu$  is  $\sigma$ -additive.

Let  $H_{\bar{\Sigma}} = \{[\alpha]_{\bar{\Sigma}} \mid \alpha \in For_{LTL}\}$ , where  $[\alpha]_{\bar{\Sigma}} = \{\sigma \in \bar{\Sigma} \mid v(w, \alpha) = 1\}$ . By  $For_{LTL}^{\circ}$  we denote the set of all LTL formulas in which  $\circ$  is the only temporal operator (i.e. there are no appearances of  $U$ ). We also introduce the set  $A = \{[\alpha] \mid \alpha \in For_{LTL}^{\circ}\}$ . Using the same argument as above, we can show that the sets  $H_{\bar{\Sigma}}$  and  $A$  are two algebras of subsets of  $\bar{\Sigma}$ . Similarly as in the definition of  $M_{T^*}$ , we define  $\mu^*$  on  $H_{\bar{\Sigma}}$  by

$$\mu^*([\alpha]_{\bar{\Sigma}}) = \sup\{r \in \mathcal{Q} \mid T^* \vdash w(\alpha) \geq r\}.$$

Reasoning as above, we conclude that  $\mu^*$  is a finitely additive measure. We also use the same symbol  $\mu^*$  to denote the restriction of  $\mu^*$  to  $A$ . We actually want to show that  $\mu^*$  is  $\sigma$ -additive on  $A$ . It is sufficient to show that if  $B = \bigcup_{n \in \omega} B_n$ , where  $B, B_n \in A$ , then there is  $n$  such that  $B = \bigcup_{n=0}^n B_n$ .

If  $2^{\mathcal{P}}$  denotes the set of subsets of  $\mathcal{P}$ , note that  $\bar{\Sigma} = 2^{\mathcal{P}} \times 2^{\mathcal{P}} \times 2^{\mathcal{P}} \times \dots$ . If we assume discrete topology on the finite set  $2^{\mathcal{P}}$  and the induced product topology on  $\bar{\Sigma}$ , then  $\bar{\Sigma}$  is a compact space as a product of compact spaces.<sup>3</sup> By definition of evaluation function  $v$ , we obtain that for every  $\alpha \in For_{LTL}^{\circ}$  there exist  $n \in \omega$  (for example  $n$  is the number of appearances of  $\circ$ ) and  $S \subseteq (2^{\mathcal{P}})^n$  such that  $[\alpha]_{\bar{\Sigma}} = S \times 2^{\mathcal{P}} \times 2^{\mathcal{P}} \times \dots$ . Note that the sets of the form  $S \times 2^{\mathcal{P}} \times 2^{\mathcal{P}} \times \dots$ , where  $S \subseteq (2^{\mathcal{P}})^n$  for some  $n \in \omega$ , are clopen (both closed and open) sets in product topology. Thus, each  $[\alpha]_{\bar{\Sigma}} \in A$  is a clopen set in  $\bar{\Sigma}$ . Now assume that  $[\alpha]_{\bar{\Sigma}} = \bigcup_{n \in \omega} [\alpha_n]_{\bar{\Sigma}}$ , where  $\alpha \in For_{LTL}^{\circ}$  and  $\alpha_n \in For_{LTL}^{\circ}$  for every  $n \in \omega$ . The set  $\{[\alpha_n]_{\bar{\Sigma}} \mid n \in \omega\}$  is an open cover of the closed subset  $[\alpha]_{\bar{\Sigma}}$  of the compact space  $\bar{\Sigma}$ , so there is a finite subcover  $\{[\alpha_{n_1}]_{\bar{\Sigma}}, \dots, [\alpha_{n_1}]_{\bar{\Sigma}}\}$  of  $[\alpha]_{\bar{\Sigma}}$ . Thus,  $\mu^*$  is  $\sigma$ -additive on  $A$ .

<sup>3</sup>For the basic notions and results about the topology used here we refer the reader to [Kechris, 1995]

Let  $F$  be the  $\sigma$ -algebra generated by  $A$ . Since  $[\alpha U \beta]_{\bar{\Sigma}} = \bigcup_{n \in \omega} [\alpha U_n \beta]_{\bar{\Sigma}}$ , we can show that  $[\alpha]_{\bar{\Sigma}} \in F$  for every  $\alpha \in For_{LTL}$ , using the induction on the number of appearances of  $U$  in  $\alpha$ . Thus,  $H_{\bar{\Sigma}} \subseteq F$ . By Caratheodory's extension theorem (see [Ash and Doléans-Dade, 1999]), there is a unique  $\sigma$ -additive probability measure  $\nu$  on  $F$  which coincide with  $\mu^*$  on  $A$ . We will actually show that  $\mu^*$  is the restriction of  $\nu$  to  $H_{\bar{\Sigma}}$ , i.e., that  $\mu^*([\alpha]_{\bar{\Sigma}}) = \nu([\alpha]_{\bar{\Sigma}})$  for all  $\alpha \in For_{LTL}$ , using the induction on the number of appearances of  $U$  in  $\alpha$ . Indeed,  $\nu([\alpha]_{\bar{\Sigma}}) = \nu(\bigcup_{n \in \omega} [\alpha U_n \beta]_{\bar{\Sigma}}) = \lim_{k \rightarrow +\infty} \nu(\bigcup_{n=1}^k [\alpha U_n \beta]_{\bar{\Sigma}}) = \lim_{k \rightarrow +\infty} \mu^*(\bigcup_{n=1}^k [\alpha U_n \beta]_{\bar{\Sigma}}) = \mu^*([\alpha U \beta]_{\bar{\Sigma}})$ . Here we used  $\sigma$ -additivity of  $\nu$ , the induction hypothesis and, in the last step, the definition of  $\mu^*$  and R6.

Thus,  $\mu^*$  is a  $\sigma$ -additive probability measure on  $H_{\bar{\Sigma}}$ . Note that we have that  $\mu^*([\alpha]_{\bar{\Sigma}}) = 1$  whenever  $T^* \vdash \alpha$ , by R3. Thus,  $\mu^*(W) = \mu^*(\bigcap_{\alpha: T^* \vdash \alpha} [\alpha]_{\bar{\Sigma}}) = 1$ , by  $\sigma$ -additivity of  $\mu^*$ .

Note that  $[\alpha] = [\alpha]_{\bar{\Sigma}} \cap W$ , so  $H \subseteq F$ . Let  $\bar{\mu}$  be the  $\sigma$ -additive probability measure on  $H$  induced by  $\mu^*$  by

$$\bar{\mu}([\alpha]) = \bar{\mu}([\alpha]_{\bar{\Sigma}} \cap W) = \mu^*([\alpha]_{\bar{\Sigma}}).$$

Note that  $\mu^*(W) = 1$  implies  $\mu^*([\alpha]_{\bar{\Sigma}}) = \mu^*([\alpha]_{\bar{\Sigma}} \cap W)$ , so  $\mu^*([\alpha]) = \nu([\alpha])$ . By definitions of  $\mu$  and  $\mu^*$  it follows that  $\mu$  and  $\nu$  coincide. Thus,  $\mu$  is  $\sigma$ -additive.

We showed that  $M_{T^*}$  is a  $PL_{LTL}$  structure. Finally, note that  $[\alpha] = [\alpha]_{M_{T^*}}$ , by the choice of  $\pi$ , so  $M_{T^*} \in PL_{LTL}^{Meas}$ .

Now we can prove the main result of this section.

### 4.3 COMPLETENESS THEOREM

**Theorem 5 (Strong completeness)** A set of formulas  $T \subseteq For$  is consistent iff it is satisfiable.

*Proof.* The direction from right to left follows from the soundness of the axiomatization  $AX_{PL_{LTL}}$ . For the other direction, we need to show that a consistent set of formulas  $T$  has a model. First we extend  $T$  to a maximal consistent set  $T^*$ , and we construct the canonical model  $M_{T^*}$ . We will show that  $M_{T^*}$  is a model of  $T^*$ , and, consequently, a model of  $T$ . It is sufficient to prove that for all  $\Phi \in For$ ,  $T^* \vdash \Phi$  iff  $M_{T^*} \models \Phi$ .

If  $\Phi = \alpha \in For_{LTL}$ . If  $\alpha \in T^*$ , then by the definition of  $W$  from  $M_{T^*}$ ,  $M_{T^*} \models \alpha$ . Conversely, if  $M_{T^*} \models \alpha$ , by Lemma 2,  $\alpha \in T^*$ .

If  $\Phi \in For_P$ , we proceed by induction on the complexity of  $\Phi$ .

Let  $\Phi = f \geq r$ . If  $f = r_1 w(\alpha_1) + \dots + r_k w(\alpha_k) + r_{k+1}$ , we can show, using the properties of supremum, that

$$r_1 \mu([\alpha_1]) + \dots + r_k \mu([\alpha_k]) + r_{k+1} = \sup\{s \mid T^* \vdash f \geq s\}.$$

If we suppose that  $f \geq r \in T^*$ , then  $r \leq \sup\{s \mid T^* \vdash f \geq s\}$ , so  $M_{T^*} \models f \geq r$ . For the other direction, assume that  $M_{T^*} \models f \geq r$ . Then  $M_{T^*} \not\models f < r$ . If  $f < r \in T^*$ ,



then, reasoning as above, we conclude  $M_{T^*} \models f < r$ , a contradiction. By Maximality of  $T^*$ , we obtain  $f \geq r \in T^*$ .

If  $\Phi = \neg\phi$ , then  $M_{T^*} \models \neg\phi$  iff  $M_{T^*} \not\models \phi$  iff  $\phi \notin T^*$  iff  $\neg\phi \in T^*$ , by maximality of  $T^*$ .

If  $\Phi = \phi \wedge \psi$ , then  $M_{T^*} \models \phi \wedge \psi$  iff  $M_{T^*} \models \phi$  and  $M_{T^*} \models \psi$  iff  $\phi, \psi \in T^*$  iff  $\phi \wedge \psi \in T^*$ , by maximality of  $T^*$ .

As it is well known, the alternative formulation of Completeness theorem, stated below, follows directly from the previous result.

**Theorem 6** *If  $T \subseteq For$  and  $\Phi \in For$ , then  $T \models \Phi$  iff  $T \vdash \Phi$ .*

## 5 THE DECIDABILITY OF $PL_{LTL}$

[Sistla and Clarke, 1985] proved that the logic LTL is decidable, and they showed that the problem of deciding whether an LTL formula is satisfiable in a path is  $PSPACE$ -complete. Note that if  $\alpha$  is not satisfiable in any path, then by Definition 6 it is not satisfiable in the logic  $PL_{LTL}$ . On the other hand, if there is a path  $\sigma$  such that  $v(\sigma, \alpha) = 1$ , then we can define a measurable structure  $M = \langle W, H, \mu, \pi \rangle$ , such that  $W = \{w\}$  is a singleton and  $\pi(w) = \sigma$  (note that in that case the range of  $\mu$  is  $\{0, 1\}$ ). Obviously,  $v(\pi(w), \alpha) = 1$  for every  $w \in W$ , so  $M \models \alpha$ . Thus, we proved that the satisfiability problem of LTL formulas for the logic  $PL_{LTL}$  is  $PSPACE$ -complete.

Now let us consider the satisfiability of a formula  $\varphi \in For_P$ . Let  $For_B(\varphi)$  denote the set of all basic probabilistic formulas which appear in  $\varphi$ . Suppose that the formula  $\varphi \in For_P$  is given in the complete disjunctive normal form (CDNF), i.e.,  $\varphi = \bigvee_{i=1}^m \varphi_i$ , where each  $\varphi_i$  is a conjunction of the formulas from  $For_B(\varphi)$  or their negations, using all elements of  $For_B(\varphi)$ , i.e. the number of conjuncts of each  $\varphi_i$  is  $|For_B(\varphi)|$ . Note that the disjunction  $\varphi$  is satisfiable iff at least one of its disjuncts  $\varphi_i$  is satisfiable.

Thus, we focus on satisfiability of the formulas of the form

$$\bigwedge_{k=1}^{|For_B(\varphi)|} \psi_k, \quad (3)$$

where each  $\psi_k$  is a basic formula or its negation. In the following, we assume that a formula of the form (3) is given, and we denote by  $F$  the set of its conjuncts  $\{\psi_k \mid k = 1, \dots, |For_B(\varphi)|\}$ .

For a LTL formula  $\alpha$ , by  $Subfor(\alpha)$  we denote the set of its subformulas. If  $For_{LTL}(F)$  is the set of all LTL formulas which appear in at least one element of  $F$  (under the scope of probability operator  $w$ ), let  $Subfor = \bigcup_{\alpha \in For_{LTL}(F)} Subfor(\alpha)$ . Let us consider the formulas

of the form

$$\bigwedge_{k=1}^{|Subfor|} \beta_k, \quad (4)$$

where each  $\beta_k$  belongs to  $Subfor \cup \{\neg\beta \mid \beta \in Subfor\}$ , and each subformula of  $\alpha$  appears exactly once (negated or not). Obviously the conjunction of any two different formulas of the form (4) is a contradiction, while the disjunction of all such formulas is a tautology. This enables us to translate the satisfiability problem to the problem of finding a solution of a system of inequalities. First, note that there are  $2^{|Subfor|}$  formulas of the form (4). First we eliminate the formulas which are not satisfiable in LTL, using the procedure from [Sistla and Clarke, 1985]. Suppose that there are  $\ell$  formulas which are satisfiable ( $\ell \leq 2^{|Subfor|}$ ). We denote those formulas by  $\alpha_1, \dots, \alpha_\ell$ .

For any formula  $\alpha \in For_{LTL}(F)$  we have that  $\alpha \in Subfor$ . Consequently,  $\alpha$  appears in each conjunction  $\alpha_k$ , negated or not. Since  $\bigvee_{k=1}^{\ell} \alpha_k$  is a tautology, there is a unique set of indices  $I_\alpha \subseteq \{1, \dots, \ell\}$  such that  $\alpha \leftrightarrow \bigvee_{i \in I_\alpha} \alpha_i$  is a tautology. Let  $\Gamma_\alpha$  be the corresponding set  $\{\alpha_i \mid i \in I_\alpha\}$ . Using the probabilistic axioms and Lemma 1(3), we obtain

$$\vdash w(\alpha) = \sum_{\alpha_i \in \Gamma_\alpha} w(\alpha_i). \quad (5)$$

Now, we can transform every formula  $\psi \in F$  of the form  $r_1 w(\gamma_1) + \dots + r_k w(\gamma_k) \geq r_{k+1}$  to the equivalent formula

$$r_1 \sum_{\alpha_i \in \Gamma_{\gamma_1}} w(\alpha_i) + \dots + r_k \sum_{\alpha_i \in \Gamma_{\gamma_k}} w(\alpha_i) \geq r_{k+1}. \quad (6)$$

Thus, we obtain that a measurable structure  $M = \langle W, H, \mu, \pi \rangle$  satisfies  $\psi$  if and only if

$$r_1 \sum_{\alpha_i \in \Gamma_{\gamma_1}} \mu([\alpha_i]) + \dots + r_k \sum_{\alpha_i \in \Gamma_{\gamma_k}} \mu([\alpha_i]) \geq r_{k+1}. \quad (7)$$

Similarly, if  $\psi$  from  $F$  is a negation of a basic probabilistic formula, then it is of the form  $r_1 w(\gamma_1) + \dots + r_k w(\gamma_k) < r_{k+1}$ , which give us the similar condition for satisfiability of  $\psi$  under  $M$ :

$$r_1 \sum_{\alpha_i \in \Gamma_{\gamma_1}} \mu([\alpha_i]) + \dots + r_k \sum_{\alpha_i \in \Gamma_{\gamma_k}} \mu([\alpha_i]) < r_{k+1}. \quad (8)$$

Let denote by  $x_i$  the probability of the formula  $\alpha_i$  in a potential model  $M = \langle W, H, \mu, \pi \rangle$  of the formula (3), i.e.,  $x_i = \mu([\alpha_i])$  each  $i \in \{1, \dots, \ell\}$ .

Let  $F_{pos}$  be the set of basic probabilistic formulas from  $F$ , and let  $F_{neg}$  be the set of formulas from  $F$  which are negations of basic probabilistic formulas. For given  $\psi \in F_{pos}$  of the form  $r_1 w(\gamma_1) + \dots + r_k w(\gamma_k) \geq r_{k+1}$  we define the inequality  $Ineq(\psi)$ , obtained by (7), as

$$Ineq(\psi) : r_1 \left( \sum_{i: \alpha_i \in \Gamma_{\gamma_1}} x_i \right) + \dots + r_k \left( \sum_{i: \alpha_i \in \Gamma_{\gamma_k}} x_i \right) \geq r_{k+1}.$$

In the same way we define  $Ineq(\psi)$  for  $\psi \in F_{neg}$  of the form  $r_1 w(\gamma_1) + \dots + r_k w(\gamma_k) < r_{k+1}$  as

$$Ineq(\psi) : r_1 \left( \sum_{i:\alpha_i \in \Gamma_{\gamma_1}} x_i \right) + \dots + r_k \left( \sum_{i:\alpha_i \in \Gamma_{\gamma_k}} x_i \right) < r_{k+1}.$$

Then the formula (3) is satisfiable iff the following sentence of the language of real closed fields is satisfiable:

$$\begin{aligned} \exists x_1 \dots \exists x_\ell \quad & \left( \bigwedge_{k=1}^{\ell} (x_k \geq 0) \right) \\ & \wedge \sum_{k=1}^{\ell} x_k = 1 \\ & \wedge \bigwedge_{\psi \in F} Ineq(\psi). \end{aligned}$$

The sentence represents a nonlinear system of linear inequalities: the first line represents non-negativity of probability measures; the second line represents the condition  $\mu(W) = \mu([\top]) = \sum_{k=1}^{\ell} \mu([\alpha_k]) = 1$ . The third line represent the conditions (7) and (8). Obviously, if the system doesn't have a solution, there is no  $\mu$  which satisfies (3). If the system has the solution  $(x_1, \dots, x_\ell) = (c_1, \dots, c_\ell)$ , then we can construct  $M = \langle W, H, \mu, \pi \rangle$  which satisfies (3) in the following way:  $W = \{w_1, \dots, w_\ell\}$ ,  $\pi(w_i)$  is any path  $\sigma$  such that  $v(\sigma, \alpha_i) = 1$ ,  $H$  is the set of all subsets of  $W$  and  $\mu$  is determined by the condition  $\mu(\{w_i\}) = c_i$ .

Since the theory of real closed fields is decidable, our logic is decidable as well. Moreover, note that the above sentence is an existential sentence. Thus, we can use Canny's decision procedure from [Canny, 1988]. Since the procedure decides satisfiability of the formula in  $PSPACE$ , we conclude that satisfiability of probabilistic formulas is in  $PSPACE$  as well.

Thus, in both probabilistic and LTL case there is a procedure which decides satisfiability of the formula in  $PSPACE$ . Since  $PSPACE$  is also a lower bound in the case of LTL formulas, we proved the following result.

**Theorem 7** *The problem of deciding whether a formula of the logic  $PL_{LTL}$  is satisfiable in a measurable structure from  $PL_{LTL}^{Meas}$  is  $PSPACE$ -complete.*

## 6 CONCLUSION

In this paper, we introduced the logic  $PL_{LTL}$  for probabilistic reasoning about temporal information. The language contains both LTL formulas and probabilistic formulas in the style of [Fagin et al., 1990], with the difference that the probabilistic operator  $w$  is now applied to LTL formulas. We propose an axiomatization for the logic and prove strong completeness. Since the semantical relationship between the operators “next” and “until” explicitly requires  $\sigma$ -additive semantics, the axiomatization contains infinitary rules of inference. We show that the satisfiability

problem is  $PSPACE$ -complete, no harder than satisfiability for LTL.

It seems that combining any standard finitary axiomatization of LTL with the axiomatization from [Fagin et al., 1990] could be extended to a weakly (but not strongly) complete axiomatization for a finitely additive restriction of our logic, which would be convenient for possible applications. On the other hand, we believe that our infinitary rules of inference can be represented using schemes (similarly as quantifiers in the first order logic are abbreviations for the infinite conjunctions and disjunctions), so that some of infinitary proofs might be finitary represented and used in automated reasoning.

Some probabilistic LTL's were motivated by the need to analyze probabilistic programs and stochastic systems [Donaldson and Gilbert, 2008, Feldman, 1984, Hansson and Jonsson, 1994, Kozen, 1985, Lehmann and Shelah, 1982]. In some of them, probabilistic operators are not explicitly mentioned in the formulas, while in the others it is possible to directly express probabilities. Our logic allows one to quantify runs satisfying some properties. In this paper we restrict our attention to theoretical issues (e.g., worst case complexity), while the possible applications (e.g., heuristic procedures for satisfiability checking) are left for the future work.

## Acknowledgements

This work was supported by the National Research Fund (FNR) of Luxembourg through project PRIMAT, and by the Serbian Ministry of Education and Science through projects ON174026 and III44006.

We wish to thank the anonymous UAI referees whose comments and suggestions helped us to improve the paper. We also wish to thank Marc van Zee for his help.

## References

- [Ash and Doléans-Dade, 1999] Ash, R. B. and Doléans-Dade, C. A. (1999). *Probability & Measure Theory, Second Edition*. Academic Press, 2 edition.
- [Canny, 1988] Canny, J. F. (1988). Some algebraic and geometric computations in  $PSPACE$ . In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 460–467.
- [Donaldson and Gilbert, 2008] Donaldson, R. and Gilbert, D. (2008). A monte carlo model checker for probabilistic ltl with numerical constraints. Technical report, University of Glasgow, Department of Computing Science.
- [Emerson, 1990] Emerson, A. E. (1990). Temporal and modal logic. pages 995–1072.

- [Emerson, 1995] Emerson, E. A. (1995). Automated temporal reasoning about reactive systems. In *Logics for Concurrency - Structure versus Automata (8th Banff Higher Order Workshop, August 27 - September 3, 1995, Proceedings)*, pages 41–101.
- [Fagin et al., 1990] Fagin, R., Halpern, J. Y., and Megiddo, N. (1990). A logic for reasoning about probabilities. *Inf. Comput.*, 87(1/2):78–128.
- [Feldman, 1984] Feldman, Y. A. (1984). A decidable propositional dynamic logic with explicit probabilities. *Information and Control*, 63(1/2):11–38.
- [Gabbay et al., 1980] Gabbay, D. M., Pnueli, A., Shelah, S., and Stavi, J. (1980). On the temporal basis of fairness. In *Conference Record of the Seventh Annual ACM Symposium on Principles of Programming Languages, Las Vegas, Nevada, USA, January 1980*, pages 163–173.
- [Grant et al., 2010] Grant, J., Parisi, F., Parker, A., and Subrahmanian, V. S. (2010). An agm-style belief revision mechanism for probabilistic spatio-temporal logics. *Artif. Intell.*, 174(1):72–104.
- [Guelev, 2000] Guelev, D. P. (2000). Probabilistic neighbourhood logic. In *Formal Techniques in Real-Time and Fault-Tolerant Systems, 6th International Symposium, FTRTFT 2000, Pune, India, September 20-22, 2000, Proceedings*, pages 264–275.
- [Haddawy, 1996] Haddawy, P. (1996). A logic of time, chance, and action for representing plans. *Artif. Intell.*, 80(1-2):243–308.
- [Halpern and Pucella, 2006] Halpern, J. Y. and Pucella, R. (2006). A logic for reasoning about evidence. *J. Artif. Intell. Res. (JAIR)*, 26:1–34.
- [Hansson and Jonsson, 1994] Hansson, H. and Jonsson, B. (1994). A logic for reasoning about time and reliability. *Formal Asp. Comput.*, 6(5):512–535.
- [Kechris, 1995] Kechris, A. S. (1995). *Classical Descriptive Set Theory (Graduate Texts in Mathematics) (v. 156)*. Springer, 1 edition.
- [Kozen, 1985] Kozen, D. (1985). A probabilistic PDL. *J. Comput. Syst. Sci.*, 30(2):162–178.
- [Lehmann and Shelah, 1982] Lehmann, D. J. and Shelah, S. (1982). Reasoning with time and chance. *Information and Control*, 53(3):165–198.
- [Marinkovic et al., 2014] Marinkovic, B., Ognjanovic, Z., Doder, D., and Perovic, A. (2014). A propositional linear time logic with time flow isomorphic to  $\omega^2$ . *J. Applied Logic*, 12(2):208–229.
- [Nilsson, 1986] Nilsson, N. J. (1986). Probabilistic logic. *Artif. Intell.*, 28(1):71–87.
- [Ognjanovic, 2006] Ognjanovic, Z. (2006). Discrete linear-time probabilistic logics: Completeness, decidability and complexity. *J. Log. Comput.*, 16(2):257–285.
- [Perovic et al., 2008] Perovic, A., Ognjanovic, Z., Raskovic, M., and Markovic, Z. (2008). A probabilistic logic with polynomial weight formulas. In *Foundations of Information and Knowledge Systems, 5th International Symposium, FoIKS 2008, Pisa, Italy, February 11-15, 2008, Proceedings*, pages 239–252.
- [Prior, 1957] Prior, A. (1957). *Time and Modality*. Clarendon Press, Oxford.
- [Reynolds, 2001] Reynolds, M. (2001). An axiomatization of full computation tree logic. *J. Symb. Log.*, 66(3):1011–1057.
- [Shakarian et al., 2011] Shakarian, P., Parker, A., Simari, G. I., and Subrahmanian, V. S. (2011). Annotated probabilistic temporal logic. *ACM Trans. Comput. Log.*, 12(2):14.
- [Sistla and Clarke, 1985] Sistla, A. P. and Clarke, E. M. (1985). The complexity of propositional linear temporal logics. *J. ACM*, 32(3):733–749.
- [van der Hoek, 1997] van der Hoek, W. (1997). Some considerations on the logic  $\text{pfd}^-$ . *Journal of Applied Non-Classical Logics*, 7(3).

---

# Training generative neural networks via Maximum Mean Discrepancy optimization

---

**Gintare Karolina Dziugaite**  
University of Cambridge

**Daniel M. Roy**  
University of Toronto

**Zoubin Ghahramani**  
University of Cambridge

## Abstract

We consider training a deep neural network to generate samples from an unknown distribution given i.i.d. data. We frame learning as an optimization minimizing a two-sample test statistic—informally speaking, a good generator network produces samples that cause a two-sample test to fail to reject the null hypothesis. As our two-sample test statistic, we use an unbiased estimate of the maximum mean discrepancy, which is the centerpiece of the nonparametric kernel two-sample test proposed by Gretton et al. [2]. We compare to the *adversarial nets* framework introduced by Goodfellow et al. [1], in which learning is a two-player game between a generator network and an adversarial discriminator network, both trained to outwit the other. From this perspective, the MMD statistic plays the role of the discriminator. In addition to empirical comparisons, we prove bounds on the generalization error incurred by optimizing the empirical MMD.

## 1 INTRODUCTION

In this paper, we consider the problem of learning generative models from i.i.d. data with unknown distribution  $\mathcal{P}$ . We formulate the learning problem as one of finding a function  $G$ , called the *generator*, such that, given an input  $Z$  drawn from some fixed *noise* distribution  $\mathcal{N}$ , the distribution of the output  $G(Z)$  is close to the data's distribution  $\mathcal{P}$ . Note that, given  $G$  and  $\mathcal{N}$ , we can easily generate new samples despite not having an explicit representation for the underlying density.

We are particularly interested in the case where the generator is a deep neural network whose parameters we must learn. Rather than being used to classify or predict, these networks transport input randomness to output random-

ness, thus inducing a distribution. The first direct instantiation of this idea is due to MacKay [7], although MacKay draws connections even further back to the work of Saund [11] and others on autoencoders, suggesting that generators can be understood as decoders. MacKay's proposal, called *density networks*, uses multi-layer perceptrons (MLP) as generators and learns the parameters by approximating Bayesian inference.

Since MacKay's proposal, there has been a great deal of progress on learning generative models, especially over high-dimensional spaces like images. Some of the most successful approaches have been based on restricted Boltzmann machines [10] and deep Boltzmann networks [3]. A recent example is the Neural Autoregressive Density Estimator due to Uria, Murray, and Larochelle [15]. An indepth survey, however, is beyond the scope of this article.

This work builds on a proposal due to Goodfellow et al. [1]. Their *adversarial nets* framework takes an indirect approach to learning deep generative neural networks: a discriminator network is trained to recognize the difference between training data and generated samples, while the generator is trained to confuse the discriminator. The resulting two-player game is cast as a minimax optimization of a differentiable objective and solved greedily by iteratively performing gradient descent steps to improve the generator and then the discriminator.

Given the greedy nature of the algorithm, Goodfellow et al. [1] give a careful prescription for balancing the training of the generator and the discriminator. In particular, two gradient steps on the discriminator's parameters are taken for every iteration of the generator's parameters. It is not clear at this point how sensitive this balance is as the data set and network vary. In this paper, we describe an approximation to adversarial learning that replaces the adversary with a closed-form nonparametric two-sample test statistic based on the Maximum Mean Discrepancy (MMD), which we adopted from the kernel two sample test [2]. We call our proposal *MMD nets*.<sup>1</sup> We give bounds on the estimation

---

<sup>1</sup>In independent work reported in a recent preprint, Li, Swer-

error incurred by optimizing an empirical estimator rather than the true population MMD and give some illustrations on synthetic and real data.

## 2 LEARNING TO SAMPLE AS OPTIMIZATION

It is well known that, for any distribution  $\mathcal{P}$  and any continuous distribution  $\mathcal{N}$  on sufficiently regular spaces  $\mathbb{X}$  and  $\mathbb{W}$ , respectively, there is a function  $G : \mathbb{W} \rightarrow \mathbb{X}$ , such that  $G(W) \sim \mathcal{P}$  when  $W \sim \mathcal{N}$ . (See, e.g., [4, Lem. 3.22].) In other words, we can transform an input from a fixed input distribution  $\mathcal{N}$  through a deterministic function, producing an output whose distribution is  $\mathcal{P}$ . For a given family  $\{G_\theta\}$  of functions  $\mathbb{W} \rightarrow \mathbb{X}$ , called *generators*, we can cast the problem of learning a generative model as an optimization

$$\arg \min_{\theta} \delta(\mathcal{P}, G_\theta(\mathcal{N})), \quad (1)$$

where  $\delta$  is some measure of discrepancy and  $G_\theta(\mathcal{N})$  is the distribution of  $G_\theta(W)$  when  $W \sim \mathcal{N}$ . In practice, we only have i.i.d. samples  $X_1, X_2, \dots$  from  $\mathcal{P}$ , and so we optimize an empirical estimate of  $\delta(\mathcal{P}, G_\theta(\mathcal{N}))$ .

### 2.1 ADVERSARIAL NETS

Adversarial nets [1] can be cast within this framework: Let  $\{D_\phi\}$  be a family of functions  $\mathbb{X} \rightarrow [0, 1]$ , called *discriminators*. We recover the adversarial nets objective with the discrepancy

$$\delta_{\text{AN}}(\mathcal{P}, G_\theta(\mathcal{N})) = \max_{\phi} E[\log D_\phi(X) + \log(1 - D_\phi(Y))],$$

where  $X \sim \mathcal{P}$  and  $Y \sim G_\theta(\mathcal{N})$ . In this case, Eq. (1) becomes

$$\min_{\theta} \max_{\phi} V(G_\theta, D_\phi)$$

where

$$V(G_\theta, D_\phi) = E[\log D_\phi(X) + \log(1 - D_\phi(G_\theta(W)))]$$

for  $X \sim \mathcal{P}$  and  $W \sim \mathcal{N}$ . The output of the discriminator  $D_\phi$  can be interpreted as the probability it assigns to its input being drawn from  $\mathcal{P}$ , and so  $V(G_\theta, D_\phi)$  is the expected log loss incurred when classifying the origin of a point equally likely to have been drawn from  $\mathcal{P}$  or  $G_\theta(\mathcal{N})$ . Therefore, optimizing  $\phi$  maximizes the probability of distinguishing samples from  $\mathcal{P}$  and  $G_\theta(\mathcal{N})$ . Assuming that the optimal discriminator exists for every  $\theta$ , the optimal generator  $G$  is that whose output distribution is closest to  $\mathcal{P}$ , as measured by the Jensen–Shannon divergence, which is minimized when  $G_\theta(\mathcal{N}) = \mathcal{P}$ .

sky, and Zemel [6] also propose to use MMD as a training objective for generative neural networks. We leave a comparison to future work.

In [1], the generators  $G_\theta$  and discriminators  $D_\phi$  are chosen to be multilayer perceptrons (MLP). In order to find a minimax solution, they propose taking alternating gradient steps along  $D_\phi$  and  $G_\theta$ . Note that the composition  $D_\phi(G_\theta(\cdot))$  that appears in the value function is yet another (larger) MLP. This fact permits the use of the back-propagation algorithm to take gradient steps.

### 2.2 MMD AS AN ADVERSARY

In their paper introducing adversarial nets, Goodfellow et al. [1] remark that a balance must be struck between optimizing the generator and optimizing the discriminator. In particular, the authors suggest  $k$  maximization steps for every one minimization step to ensure that  $D_\phi$  is well synchronized with  $G_\theta$  during training. A large value for  $k$ , however, can lead to overfitting. In their experiments, for every step taken along the gradient with respect to  $G_\theta$ , they take two gradient steps with respect to  $D_\phi$  to bring  $D_\phi$  closer to the desired optimum (Goodfellow, pers. comm.).

It is unclear how sensitive this balance is. Regardless, while adversarial networks deliver impressive sampling performance, the optimization takes approximately 7.5 hours to train on the MNIST dataset running on a GeForce GTX TITAN GPU from nVidia with 6GB RAM. Can we potentially speed up the process with a more tractable choice of adversary?

Our proposal is to replace the adversary with the kernel two-sample test introduced by Gretton et al. [2]. In particular, we replace the family of discriminators with a family  $\mathcal{H}$  of test functions  $\mathbb{X} \rightarrow \mathbb{R}$ , closed under negation, and use the maximum mean discrepancy between  $\mathcal{P}$  and  $G_\theta(\mathcal{N})$  over  $\mathcal{H}$ , given by

$$\delta_{\text{MMD}_{\mathcal{H}}}(\mathcal{P}, G_\theta(\mathcal{N})) = \sup_{f \in \mathcal{H}} E[f(X)] - E[f(Y)], \quad (2)$$

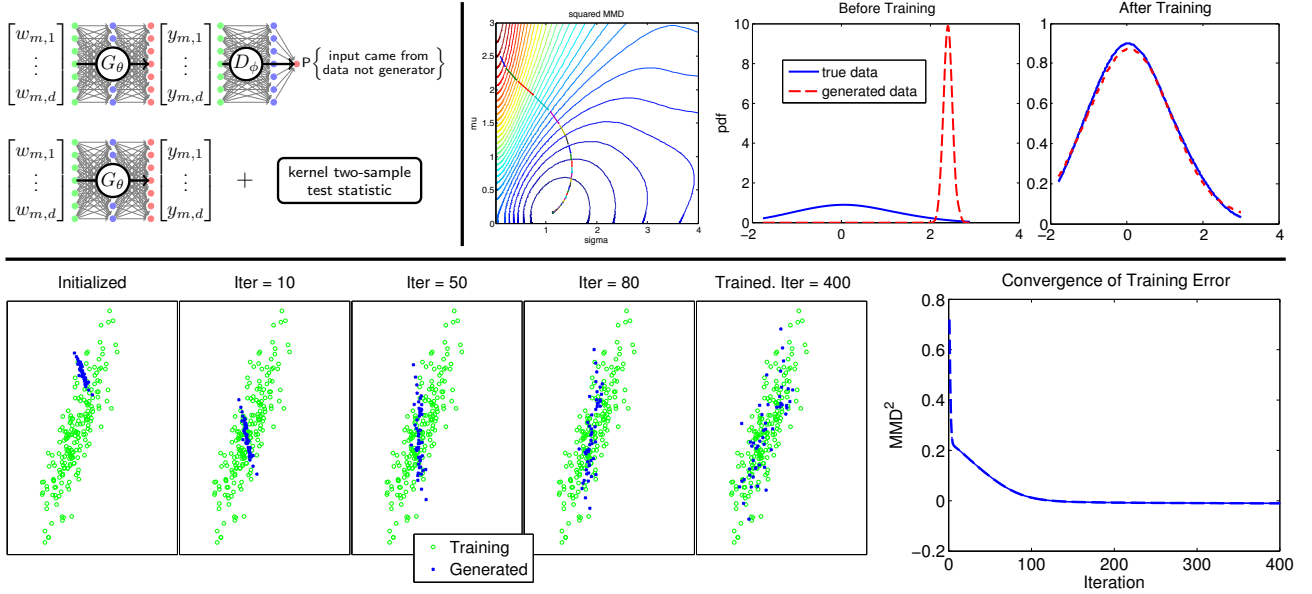
where  $X \sim \mathcal{P}$  and  $Y \sim G_\theta(\mathcal{N})$ . See Fig. 1 for a comparison of the architectures of adversarial and MMD nets.

While Eq. (2) involves a maximization over a family of functions, Gretton et al. [2] show that it can be solved in closed form when  $\mathcal{H}$  is a reproducing kernel Hilbert space (RKHS).

More carefully, let  $\mathcal{H}$  be a reproducing kernel Hilbert space (RKHS) of real-valued functions on  $\Omega$  and let  $\langle \cdot, \cdot \rangle_{\mathcal{H}}$  denote its inner product. By the reproducing property it follows that there exists a *reproducing kernel*  $k \in \mathcal{H}$  such that every  $f \in \mathcal{H}$  can be expressed as

$$f(x) = \langle f, k(\cdot, x) \rangle_{\mathcal{H}} = \sum \alpha_i k(x, x_i) \quad (3)$$

The functions *induced by a kernel*  $k$  are those functions in the closure of the span of the set  $\{k(\cdot, x) : x \in \Omega\}$ , which is necessarily an RKHS. Note, that for every positive definite kernel there is a unique RKHS  $\mathcal{H}$  such that every function in  $\mathcal{H}$  satisfies Eq. (3).



**Figure 1:** (top left) Comparison of adversarial nets and MMD nets. (top right) Here we present a simple one-dimensional illustration of optimizing a generator via MMD. Both the training data and noise data are Gaussian distributed and we consider the class of generators given by  $G_{(\mu, \sigma)}(w) = \mu + \sigma w$ . The plot on the left shows the isocontours of the MMD-based cost function and the path taken by gradient descent. On right, we show the distribution of the generator before and after a number of training iterations, as compared with the data generating distribution. Here we did not resample the generated points and so we do not expect to be able to drive the MMD to zero and match the distribution exactly. (bottom) The same procedure is repeated here for a two-dimensional dataset. On the left, we see the gradual alignment of the Gaussian-distributed input data to the Gaussian-distributed output data as the parameters of the generator  $G_\theta$  are optimized. The learning curve on the right shows the decrease in MMD obtained via gradient descent.

Assume that  $\mathbb{X}$  is a nonempty compact metric space and  $\mathcal{F}$  a class of functions  $f : \mathbb{X} \rightarrow \mathbb{R}$ . Let  $p$  and  $q$  be Borel probability measures on  $\mathbb{X}$ , and let  $X$  and  $Y$  be random variables with distribution  $p$  and  $q$ , respectively. The *maximum mean discrepancy* (MMD) between  $p$  and  $q$  is

$$\text{MMD}(\mathcal{F}, p, q) = \sup_{f \in \mathcal{F}} E[f(X)] - E[f(Y)]$$

If  $\mathcal{F}$  is chosen to be an RKHS  $\mathcal{H}$ , then

$$\text{MMD}^2(\mathcal{F}, p, q) = \|\mu_p - \mu_q\|_{\mathcal{H}}^2$$

where  $\mu_p \in \mathcal{H}$  is the *mean embedding* of  $p$ , given by

$$\mu_p = \int_{\mathbb{X}} k(x, \cdot) p(dx) \in \mathcal{H}$$

and satisfying, for all  $f \in \mathcal{H}$ ,

$$E[f(X)] = \langle f, \mu_p \rangle_{\mathcal{H}}.$$

The properties of  $\text{MMD}(\mathcal{H}, \cdot, \cdot)$  depend on the underlying RKHS  $\mathcal{H}$ . For our purposes, it suffices to say that if we take  $\mathbb{X}$  to be  $\mathbb{R}^D$  and consider the RKHS  $\mathcal{H}$  induced by Gaussian or Laplace kernels, then MMD is a metric, and so the minimum of our learning objective is achieved uniquely by  $\mathcal{P}$ , as desired. (For more details, see Sriperumbudur et al. [12].)

In practice, we often do not have access to  $p$  or  $q$ . Instead, we are given independent i.i.d. data  $X, X', X_1, \dots, X_N$  and  $Y, Y', Y_1, \dots, Y_M$  from  $p$  and  $q$ , respectively, and would like to estimate the MMD. Gretton et al. [2] showed that

$$\text{MMD}^2[\mathcal{H}, p, q] = E[k(X, X') - 2k(X, Y) + k(Y, Y')]$$

and then proposed an unbiased estimator

$$\begin{aligned} \text{MMD}_u^2[\mathcal{H}, X, Y] &= \frac{1}{N(N-1)} \sum_{n \neq n'} k(x_n, x_{n'}) \\ &+ \frac{1}{M(M-1)} \sum_{m \neq m'} k(y_m, y_{m'}) \\ &- \frac{2}{MN} \sum_{m=1}^M \sum_{n=1}^N k(x_n, y_m). \end{aligned} \quad (4)$$

### 3 MMD NETS

With an unbiased estimator of the MMD objective in hand, we can now define our proposal, *MMD nets*: Fix a neural network  $G_\theta$ , where  $\theta$  represents the parameters of the network. Let  $W = (w_1, \dots, w_M)$  denote noise inputs drawn from  $\mathcal{N}$ , let  $Y_\theta = (y_1, \dots, y_m)$  with  $y_j = G_\theta(w_j)$  denote

---

**Algorithm 1** Stochastic gradient descent for MMD nets.

---

Initialize  $M, \theta, \alpha, k$   
 Randomly divide training set  $X$  into  $N_{\text{mini}}$  mini batches  
**for**  $i \leftarrow 1$ , number-of-iterations **do**  
   Regenerate noise inputs  $\{w_i\}_{i=1, \dots, M}$  every  $r$  iterations  
   **for**  $n_{\text{mini}} \leftarrow 1, N_{\text{mini}}$  **do**  
     **for**  $m \leftarrow 1, M$  **do**  
        $y_m \leftarrow G_\theta(w_m)$   
     **end for**  
     compute the  $n$ 'th minibatch's gradient  $\nabla C^{(n)}$   
     update learning rate  $\alpha$  (e.g., RMSPROP)  
      $\theta \leftarrow \theta - \alpha \nabla C_n$   
   **end for**  
**end for**

---

the noise inputs transformed by the network  $G_\theta$ , and let  $X = (x_1, \dots, x_N)$  denote the training data in  $\mathbb{R}^D$ . Given a positive definite kernel  $k$  on  $\mathbb{R}^D$ , we minimize  $C(Y_\theta, X)$  as a function of  $\theta$ , where

$$C(Y_\theta, X) = \frac{1}{M(M-1)} \sum_{m \neq m'} k(y_m, y_{m'}) - \frac{2}{MN} \sum_{m=1}^M \sum_{n=1}^N k(y_m, x_n).$$

Note that  $C(Y_\theta, X)$  is composed of only those parts of the unbiased estimator (Eq. (4)) that depend on  $\theta$ .

In practice, the minimization is solved by gradient descent, possibly on subsets of the data. More carefully, the chain rule gives us

$$\nabla C(Y_\theta, X) = \frac{1}{N} \sum_{n=1}^N \sum_{m=1}^M \frac{\partial C_n(Y_\theta, X_n)}{\partial y_m} \frac{\partial G_\theta(w_m)}{\partial \theta},$$

where

$$C_n(Y_\theta, X_n) = \frac{1}{M(M-1)} \sum_{m \neq m'} k(y_m, y_{m'}) - \frac{2}{M} \sum_{m=1}^M k(y_m, x_n).$$

Each derivative  $\frac{\partial C_n(Y_\theta, X_n)}{\partial y_m}$  is easily computed for standard kernels like the RBF kernel. Our gradient  $\nabla C(Y_\theta, X_n)$  depends on the partial derivatives of the generator with respect to its parameters, which we can compute using back propagation.

## 4 MMD GENERALIZATION BOUNDS

MMD nets operate by minimizing an empirical estimate of the MMD. This estimate is subject to Monte Carlo error and so the network weights (parameters)  $\hat{\theta}$  that are found to

minimize the empirical MMD may do a poor job at minimizing the exact population MMD. We show that, for sufficiently large data sets, this estimation error is bounded, despite the space of parameters  $\theta$  being continuous and high dimensional.

Let  $\Theta$  denote the space of possible parameters for the generator  $G_\theta$ , let  $\mathcal{N}$  be the distribution on  $\mathcal{W}$  for the noisy inputs, and let  $p_\theta = G_\theta(\mathcal{N})$  be the distribution of  $G_\theta(W)$  when  $W \sim \mathcal{N}$  for  $\theta \in \Theta$ . Let  $\hat{\theta}$  be the value optimizing the unbiased empirical MMD estimate, i.e.,

$$\text{MMD}_u^2(\mathcal{H}, X, Y_{\hat{\theta}}) = \inf_{\theta} \text{MMD}_u^2(\mathcal{H}, X, Y_\theta), \quad (5)$$

and let  $\theta^*$  be the value optimizing the population MMD, i.e.,

$$\text{MMD}^2(\mathcal{H}, p_{\text{data}}, p_{\theta^*}) = \inf_{\theta} \text{MMD}^2(\mathcal{H}, p_{\text{data}}, p_\theta).$$

We are interested in bounding the difference

$$\text{MMD}^2(\mathcal{H}, p_{\text{data}}, p_{\hat{\theta}}) - \text{MMD}^2(\mathcal{H}, p_{\text{data}}, p_{\theta^*}).$$

To that end, for a measured space  $\mathcal{X}$ , write  $L_\infty(\mathcal{X})$  for the space of essentially bounded functions on  $\mathcal{X}$  and write  $B(L_\infty(\mathcal{X}))$  for the unit ball under the sup norm, i.e.,

$$B(L_\infty(\mathcal{X})) = \{f: \mathcal{X} \rightarrow \mathbb{R} : (\forall x \in \mathcal{X}) f(x) \in [-1, 1]\}.$$

The bounds we obtain will depend on a notion of complexity captured by the fat-shattering dimension:

**Definition 1** (Fat-shattering [8]). Let  $\mathcal{X}_N = \{x_1, \dots, x_N\} \subset \mathcal{X}$  and  $\mathcal{F} \subset B(L_\infty(\mathcal{X}))$ . For every  $\varepsilon > 0$ ,  $\mathcal{X}_N$  is said to be  $\varepsilon$ -shattered by  $\mathcal{F}$  if there is some function  $h: \mathcal{X} \rightarrow \mathbb{R}$ , such that for every  $I \subset \{1, \dots, N\}$  there is some  $f_I \in \mathcal{F}$  for which

$$\begin{aligned} f_I(x_n) &\geq h(x_n) + \varepsilon \text{ if } n \in I, \\ f_I(x_n) &\leq h(x_n) - \varepsilon \text{ if } n \notin I. \end{aligned}$$

For every  $\varepsilon$ , the *fat-shattering dimension* of  $\mathcal{F}$ , written  $\text{fat}_\varepsilon(\mathcal{F})$ , is defined as

$$\text{fat}_\varepsilon(\mathcal{F}) = \sup \{|\mathcal{X}_N| : \mathcal{X}_N \subset \mathcal{X}, \mathcal{X}_N \text{ is } \varepsilon\text{-shattered by } \mathcal{F}\}.$$

Consider the class

$$\mathcal{G}_{k+}^{\mathbb{X}} = \{g = k(x, G_\theta(\cdot)) : x \in \mathbb{X}, \theta \in \Theta\}$$

of functions from  $\mathcal{W}$  to  $\mathbb{R}$  that are compositions of some generator and the kernel with some fixed input, and the (sub)class

$$\mathcal{G}_{k+} = \{g = k(G_\theta(w), G_\theta(\cdot)) : w \in \mathcal{W}, \theta \in \Theta\}.$$

We then have the following bound on the estimation error:

**Theorem 1** (estimation error). *Assume the kernel is bounded by one and that there exists  $\gamma_1, \gamma_2 > 1$  and  $p_1, p_2 \in \mathbb{N}$  such that, for all  $\varepsilon > 0$ , it holds that  $\text{fat}_\varepsilon(\mathcal{G}_{k+}) \leq \gamma_1 \varepsilon^{-p_1}$  and  $\text{fat}_\varepsilon(\mathcal{G}_{k+}^X) \leq \gamma_2 \varepsilon^{-p_2}$ . Then with probability at least  $1 - \delta$ ,*

$$\text{MMD}^2(\mathcal{H}, p_{\text{data}}, p_{\hat{\theta}}) < \text{MMD}^2(\mathcal{H}, p_{\text{data}}, p_{\theta^*}) + \varepsilon,$$

with

$$\varepsilon = r(p_1, \gamma_1, M) + r(p_2, \gamma_2, M - 1) + 12M^{-\frac{1}{2}} \sqrt{\log \frac{2}{\delta}},$$

where the rate  $r(p, \gamma, M)$  is

$$r(p, \gamma, M) = C_p \sqrt{\gamma} \begin{cases} M^{-\frac{1}{2}} & \text{if } p < 2, \\ M^{-\frac{1}{2}} \log^{\frac{3}{2}}(M) & \text{if } p = 2, \\ M^{-\frac{1}{p}} & \text{if } p > 2, \end{cases}$$

for constants  $C_{p_1}$  and  $C_{p_2}$  depending on  $p_1$  and  $p_2$  alone.

The proof appears in the appendix. We can obtain simpler, but slightly more restrictive, hypotheses if we bound the fat-shattering dimension of the class of generators  $\{G_\theta : \theta \in \Theta\}$  alone: Take the observation space  $\mathbb{X}$  to be a bounded subset of a finite-dimensional Euclidean space and the kernel to be Lipschitz continuous and translation invariant. For the RBF kernel, the Lipschitz constant is proportional to the inverse of the length-scale: the resulting bound loosens as the length scale shrinks.

## 5 EMPIRICAL EVALUATION

In this section, we demonstrate the approach on an illustrative synthetic example as well as the standard MNIST digits and Toronto Face Dataset (TFD) benchmarks. We show that MMD-based optimization of the generator rapidly delivers a generator that produces recognizable samples, but these samples are inferior to those produced by adversarial networks, both visually and as measured by an estimate of the mean log density on a held-out test set.

### 5.1 GAUSSIAN DATA, KERNEL, AND GENERATOR

Under an RBF kernel and Gaussian generator with parameters  $\theta = \{\mu, \sigma\}$ , it is straightforward to find the gradient of  $C(Y_\theta, X)$  by applying the chain rule. Using fixed random standard normal numbers  $\{w_1, \dots, w_M\}$ , we have  $y_m = \mu + \sigma w_m$  for  $m \in \{1, \dots, M\}$ . The result of these illustrative synthetic experiments can be found in Fig. 1. The dataset consisted of  $N = 200$  samples from a standard normal and  $M = 50$  noise input samples were generated from a standard normal with a fixed random seed. The algorithm was initialized at values  $\{\mu, \sigma\} = \{2.5, 0.1\}$ . We fixed the learning rate to 0.5 and ran gradient descent steps for  $K = 250$  iterations.

### 5.2 MNIST DIGITS

We evaluated MMD nets on MNIST digits [5]. The generator was chosen to be a fully connected, 3 hidden layer neural network with sigmoidal activation functions. Following Gretton et al. [2], we used a radial basis function (RBF) kernel, but also evaluated the rational quadratic (RQ) kernel [9] and Laplacian kernel, but found that the RBF performed best in the parameter ranges we evaluated. We used Bayesian optimization (WHETLab) to set the bandwidth of the RBF and the number of neurons in each layer on initial test runs of 50,000 iterations. However, one can get a similar-quality generator simply using the median heuristic [2] to set the kernel bandwidth. The learning rate was adjusting during optimization by RMSPROP [14].

Fig. 2 presents the digits learned after 1,000,000 iterations. (Doubling the number of iterations produced similar images.) We performed minibatch stochastic gradient descent, resampling the generated digits every 300 iterations, with minibatches of 500 training and generated points. It is clear that the digits produced have many artifacts not appearing in the MNIST data set. Indeed, the mean log density of held-out test data was estimated to be only  $113 \pm 2$ , as compared with the reported  $225 \pm 2$  achieved by adversarial nets. On the other hand, most of the gain is achieved by MMD nets in the first 100-200k iterations, and so perhaps MMD nets could be used to initialize a network further optimized by other means.

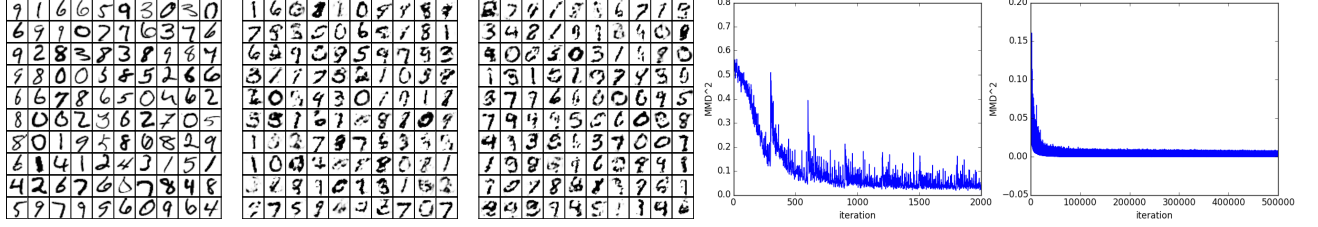
### 5.3 TORONTO FACE DATASET

We also evaluated MMD nets on the Toronto face dataset (TFD) [13]. We used a 3-hidden-layer sigmoidal MLP with similar architecture (1000, 600, and 1000 units) and RBF kernel for the cost function with the same hyper parameter. We used 500 training and generated points per batch. The generated points were resampled every 500 iterations. The network was optimized for 500,000 iterations. Samples from the resulting network are plotted in Fig. 3. Again, the samples produced by MMD nets are clearly distinguishable from the training samples and this is reflected in a much lower mean log density than adversarial nets.

## 6 CONCLUSION

MMD offers a closed-form surrogate for the discriminator in the adversarial nets framework. After using Bayesian optimization for the parameters, we found that the network produced samples that were visually similar, but far from indistinguishable from those used to train the network. On one hand, adversarial nets handily outperformed MMD nets in terms of mean log density. On the other, MMD nets achieve most of their gain quickly and so it seems promising to combine MMD nets with another technique, perhaps using MMD nets to initialize a more costly procedure.





**Figure 2:** (top-left) MNIST digits from the training set. (top-right) Newly generated digits produced after 1,000,000 iterations (approximately 5 hours). Despite the remaining artifacts, the resulting kernel-density estimate of the test data is state of the art. (top-center) Newly generated digits after 300 further iterations optimizing the associated empirical MMD. (bottom-left) MMD learning curves for first 2000 iterations. (bottom-right) MMD learning curves from 2000 to 500,000 iterations. Note the difference in y-axis scale. No appreciable change is seen in later iterations.



**Figure 3:** (left) TFD. (right) Faces generated by network trained for 500,000 iterations. (center) After an additional 500 iterations.

## A PROOFS

We begin with some preliminaries and known results:

**Definition 2** ([8]). A random variable  $\sigma$  is said to be a *Rademacher random variable* if it takes values in  $\{-1, 1\}$ , each with probability  $1/2$ .

**Definition 3** ([8]). Let  $\mu$  be a probability measure on  $\mathcal{X}$ , and let  $\mathcal{F}$  be a class of uniformly bounded functions on  $\mathcal{X}$ . Then the *Rademacher complexity* of  $\mathcal{F}$  (with respect to  $\mu$ ) is

$$R_N(\mathcal{F}) = E_{\mu} E_{\sigma_1, \dots, \sigma_N} \left[ \frac{1}{\sqrt{N}} \sup_{f \in \mathcal{F}} \left| \sum_{n=1}^N \sigma_n f(X_n) \right| \right],$$

where  $\sigma = (\sigma_1, \sigma_2, \dots)$  is a sequence of independent Rademacher random variables, and  $X_1, X_2, \dots$  are independent,  $\mu$ -distributed random variables, independent also from  $\sigma$ .

**Theorem 2** (McDiarmids Inequality [8]). *Let  $f : \mathcal{X}_1 \times \dots \times \mathcal{X}_N \rightarrow \mathbb{R}$  and assume there exists  $c_1, \dots, c_N \geq 0$  such that, for all  $k \in \{1, \dots, N\}$ , we have*

$$\sup_{x_1, \dots, x_k, x'_k, \dots, x_N} |f(x_1, \dots, x_k, \dots, x_N) - f(x_1, \dots, x'_k, \dots, x_N)| \leq c_k.$$

*Then, for all  $\varepsilon > 0$  and independent random variables  $\xi_1, \dots, \xi_N$  in  $\mathcal{X}$ ,*

$$\Pr \{f(\xi_1, \dots, \xi_N) - E(f(\xi_1, \dots, \xi_N)) \geq \varepsilon\} < \exp \left( \frac{-2\varepsilon^2}{\sum_{n=1}^N c_n^2} \right).$$

**Theorem 3** ([8, Thm. 2.35]). *Let  $\mathcal{F} \subset B(L_{\infty}(\mathcal{X}))$ . Assume there exists  $\gamma > 1$ , such that for all  $\varepsilon > 0$ ,  $\text{fat}_{\varepsilon}(\mathcal{F}) \leq \gamma \varepsilon^{-p}$  for some  $p \in \mathbb{N}$ . Then there exists constants  $C_p$  depending on  $p$  only, such that  $R_N(\mathcal{F}) \leq C_p \Psi(p, N, \gamma)$  where*

$$\Psi(p, N, \gamma) = \gamma^{\frac{1}{2}} \begin{cases} 1 & \text{if } 0 < p < 2 \\ \log^{\frac{3}{2}} N & \text{if } p = 2 \\ N^{\frac{1}{2} - \frac{1}{p}} & \text{if } p > 2. \end{cases}$$

**Theorem 4** ([2]). *Assume  $0 \leq k(x_i, x_j) \leq K$ ,  $M = N$ . Then*

$$\Pr [|\text{MMD}_u^2(\mathcal{H}, X, Y_{\theta}) - \text{MMD}^2(\mathcal{H}, p_{\text{data}}, p_{\theta})| > \varepsilon] \leq \delta_{\varepsilon}$$

where

$$\delta_{\varepsilon} = 2 \exp \left( -\frac{\varepsilon^2 M}{16K^2} \right).$$

The case where  $\Theta$  is a finite set is elementary:

**Theorem 5** (estimation error for finite parameter set). *Let  $p_{\theta}$  be the distribution of  $G_{\theta}(W)$ , with  $\theta$  taking values in some finite set  $\Theta = \{\theta_1, \dots, \theta_T\}$ ,  $T < \infty$ . Then, with probability at least  $1 - (T + 1)\delta_{\varepsilon}$ , where  $\delta_{\varepsilon}$  is defined as in Theorem 4, we have*

$$\text{MMD}^2(\mathcal{H}, p_{\text{data}}, p_{\hat{\theta}}) < \text{MMD}^2(\mathcal{H}, p_{\text{data}}, p_{\theta^*}) + 2\varepsilon.$$

*Proof.* Let  $\mathcal{E}(\theta) = \text{MMD}_u^2(\mathcal{H}, X, Y_{\theta})$  and let  $\mathcal{T}(\theta) = \text{MMD}^2(\mathcal{H}, p_{\text{data}}, p_{\theta})$ .

Note, that the upper bound stated in Theorem 4 holds for the parameter value  $\theta^*$ , i.e.,

$$\Pr [|\mathcal{E}(\theta^*) - \mathcal{T}(\theta^*)| > \varepsilon] \leq \delta_{\varepsilon}. \quad (6)$$

Because  $\hat{\theta}$  depends on the training data  $X$  and generator data  $Y$ , we use a uniform bound that holds over all  $\theta$ . Specifically,

$$\begin{aligned} \Pr \left[ |\mathcal{E}(\hat{\theta}) - \mathcal{T}(\hat{\theta})| > \varepsilon \right] &\leq \Pr \left[ \sup_{\theta} |\mathcal{E}(\theta) - \mathcal{T}(\theta)| > \varepsilon \right] \\ &\leq \sum_{t=1}^T \Pr \left[ |\mathcal{E}(\hat{\theta}) - \mathcal{T}(\hat{\theta})| > \varepsilon \right] \leq T\delta_\varepsilon. \end{aligned} \quad (7)$$

This yields that with probability at least  $1 - T\delta_\varepsilon$ ,

$$\begin{aligned} 2\varepsilon &\geq |\mathcal{E}(\hat{\theta}) - \mathcal{T}(\hat{\theta})| + |\mathcal{E}(\theta^*) - \mathcal{T}(\theta^*)| \\ &\geq |\mathcal{E}(\theta^*) - \mathcal{E}(\hat{\theta}) + \mathcal{T}(\hat{\theta}) - \mathcal{T}(\theta^*)|. \end{aligned} \quad (8)$$

Since  $\theta^*$  was chosen to minimize  $\mathcal{T}(\theta)$ , we know that  $\mathcal{T}(\hat{\theta}) \geq \mathcal{T}(\theta^*)$ . Similarly, by Eq. (5),  $\mathcal{E}(\theta^*) \geq \mathcal{E}(\hat{\theta})$ . Therefore it follows that

$$\begin{aligned} 2\varepsilon &\geq \mathcal{T}(\hat{\theta}) - \mathcal{T}(\theta^*) \\ &= \text{MMD}^2(\mathcal{H}, p_{\text{data}}, p_{\theta^*}) - \text{MMD}^2(\mathcal{H}, p_{\text{data}}, p_{\hat{\theta}}) \end{aligned}$$

proving the theorem.  $\square$

**Corollary 1.** *With probability at least  $1 - \delta$ ,*

$$\text{MMD}^2(\mathcal{H}, p_{\text{data}}, p_{\hat{\theta}}) < \text{MMD}^2(\mathcal{H}, p_{\text{data}}, p_{\theta^*}) + 2\varepsilon_\delta,$$

where

$$\varepsilon_\delta = 8K \sqrt{\frac{1}{M} \log [2(T+1)\delta]}.$$

In order to prove the general result, we begin with some technical lemmas. The development here owes much to Gretton et al. [2].

**Lemma 1.** *Let  $\mathcal{F} = \{f : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}\}$  and*

$$\mathcal{F}_+ = \{h = f(y, \cdot) : f \in \mathcal{F}, y \in \mathcal{Y}\} \cap B(L_\infty(\mathcal{Y})).$$

*Let  $\{Y_n\}_{n=1}^N$  be  $\mu$ -distributed independent random variables in  $\mathcal{Y}$ . Assume for some  $\gamma > 1$  and some  $p \in \mathbb{N}$ , we have  $\text{fat}_\varepsilon(\mathcal{F}_+) \leq \gamma\varepsilon^{-p}$ , for all  $\varepsilon > 0$ . For  $y_n \in \mathcal{Y} \quad \forall n = 1, \dots, N$ , define  $\rho(y_1, \dots, y_N)$  to be*

$$\sup_{f \in \mathcal{F}} \left| E(f(Y, Y')) - \frac{1}{N(N-1)} \sum_{n \neq n'} f(y_n, y_{n'}) \right|.$$

*Then there exists a constant  $C$  that depends on  $p$ , such that*

$$E(\rho(Y_1, \dots, Y_N)) \leq \frac{C}{\sqrt{N-1}} \Psi(\gamma, N-1, p).$$

*Proof.* Let us introduce  $\{\zeta_n\}_{n=1}^N$ , where  $\zeta_n$  and  $Y_{n'}$  have the same distribution and are independent for all  $n, n' \in \{1, \dots, N\}$ . Then the following is true:

$$E(f(Y, Y')) = E\left(\frac{1}{N(N-1)} \sum_{n, n': n \neq n'} f(\zeta_n, \zeta_{n'})\right)$$

Using Jensen's inequality and the independence of  $Y, Y'$  and  $Y_n, Y_{n'}$ , we have

$$\begin{aligned} &E(\rho(Y_1, \dots, Y_N)) \\ &= E\left(\sup_{f \in \mathcal{F}} \left| E(f(Y, Y')) \right. \right. \\ &\quad \left. \left. - \frac{1}{N(N-1)} \sum_{n \neq n'} f(Y_n, Y_{n'}) \right| \right) \\ &\leq E\left(\sup_{f \in \mathcal{F}} \left| \frac{1}{N(N-1)} \sum_{n \neq n'} f(\zeta_n, \zeta_{n'}) \right. \right. \\ &\quad \left. \left. - \frac{1}{N(N-1)} \sum_{n \neq n'} f(Y_n, Y_{n'}) \right| \right). \end{aligned} \quad (9)$$

Introducing conditional expectations allows us to rewrite the equation with the sum over  $n$  outside the expectations. I.e., Eq. (9) equals

$$\begin{aligned} &\frac{1}{N} \sum_n E E^{(Y_n, \zeta_n)} \left( \sup_{f \in \mathcal{F}} \left| \frac{1}{N-1} \sum_{n \neq n'} \Phi(\zeta_n, \zeta_{n'}, Y_n, Y_{n'}) \right| \right) \\ &= E E^{(Y, \zeta)} \left( \sup_{f \in \mathcal{F}} \left| \frac{1}{N-1} \sum_{n=1}^{N-1} \sigma_n \Phi(\zeta, \zeta_n, Y, Y_n) \right| \right), \end{aligned} \quad (10)$$

where  $\Phi(x, x', y, y') = f(x, x') - f(y, y')$ . The second equality follows by symmetry of random variables  $\{\zeta_n\}_{n=1}^{N-1}$ . Note that we also added Rademacher random variables  $\{\sigma_n\}_{n=1}^{N-1}$  before each term in the sum since  $(f(\zeta_n, \zeta_{n'}) - f(Y_n, Y_{n'}))$  has the same distribution as  $-(f(\zeta_n, \zeta_{n'}) - f(Y_n, Y_{n'}))$  for all  $n, n'$  and therefore the  $\sigma$ 's do not affect the expectation of the sum.

Note that  $\zeta_m$  and  $Y_m$  are identically distributed. Thus the triangle inequality implies that Eq. (10) is less than or equal to

$$\begin{aligned} &\frac{2}{N-1} E \left( E^{(Y)} \left( \sup_{f \in \mathcal{F}} \left| \sum_{n=1}^{N-1} \sigma_n f(Y, Y_n) \right| \right) \right) \\ &\leq \frac{2}{\sqrt{N-1}} R_{N-1}(\mathcal{F}_+), \end{aligned}$$

where  $R_{N-1}(\mathcal{F}_+)$  is the Rademacher's complexity of  $\mathcal{F}_+$ . Then by Theorem 3, we have

$$E(\rho(Y_1, \dots, Y_N)) \leq \frac{C}{\sqrt{N-1}} \Psi(\gamma, N-1, p).$$

$\square$

**Lemma 2.** *Let  $\mathcal{F} = \{f : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}\}$  and  $\mathcal{F}_+ = \{f : x \times \mathcal{Y} \rightarrow \mathbb{R}, x \in \mathcal{X}\}$  and assume  $\mathcal{F}_+ \subset B(L_\infty(\mathcal{Y}))$ . Let  $\{X_n\}_{n=1}^N$  and  $\{Y_m\}_{m=1}^M$  be  $\nu$ - and  $\mu$ -distributed independent random variables in  $\mathcal{X}$  and  $\mathcal{Y}$ , respectively. Assume for some  $\gamma > 1$ , such that for all  $\varepsilon > 0$ ,  $\text{fat}_\varepsilon(\mathcal{F}_+) \leq \gamma\varepsilon^{-p}$ ,*

for some  $p \in \mathbb{N}$ . For all  $x_n \in \mathcal{X}$ ,  $n \leq N$ , and all  $y_m \in \mathcal{Y}$ ,  $m \leq M$ , define

$$\rho(x_1, \dots, x_N, y_1, \dots, y_M) = \sup_{f \in \mathcal{F}} \left| E(f(X, Y)) - \frac{1}{NM} \sum_{n,m} f(x_n, y_m) \right|.$$

Then there exists  $C$  that depends on  $p$ , such that

$$E(\rho(X_1, \dots, X_N, Y_1, \dots, Y_M)) \leq \frac{C}{\sqrt{M}} \Psi(\gamma, M, p).$$

*Proof.* The proof is very similar to that of Lemma 1.  $\square$

*Proof of Theorem 1.* The proof follows the same steps as the proof of Theorem 5 apart from a stronger uniform bound stated in Eq. (7). I.e., we need to show:

$$\Pr \left[ \sup_{\theta \in \Theta} |\mathcal{E}(\theta) - \mathcal{T}(\theta)| \geq \varepsilon \right] \leq \delta.$$

Expanding MMD as defined by Eq. (4), and substituting  $Y = G_\theta(W)$ , yields

$$\begin{aligned} & \sup_{\theta \in \Theta} |\mathcal{E}(\theta) - \mathcal{T}(\theta)| \\ &= \sup_{\theta \in \Theta} \left| E(k(X, X')) \right. \\ & \quad - \frac{1}{N(N-1)} \sum_{n' \neq n} k(X_n, X_{n'}) \\ & \quad + E(k(G_\theta(W), G_\theta(W'))) \\ & \quad - \frac{1}{M(M-1)} \sum_{m \neq m'} k(G_\theta(W_m), G_\theta(W_{m'})) \\ & \quad - 2E(k(X, G_\theta(W))) \\ & \quad \left. + \frac{2}{MN} \sum_{m,n} k(X_n, G_\theta(W_m)) \right|. \end{aligned} \quad (11)$$

For all  $n \in \{1, \dots, N\}$ ,  $k(X_n, X_{n'})$  does not depend on  $\theta$  and therefore the first two terms of the equation above can be taken out of the supremum. Also, note that since  $|k(\cdot, \cdot)| \leq K$ , we have

$$\left| \zeta(x_1, \dots, x_n, \dots, x_N) - \zeta(x_1, \dots, x'_n, \dots, x_N) \right| \leq \frac{2K}{N},$$

where

$$\zeta(x_1, \dots, x_N) = \frac{1}{N(N-1)} \sum_{n, n': n' \neq n} k(x_n, x_{n'}),$$

and  $\zeta$  is an unbiased estimate of  $E(k(X, X'))$ . Then from McDiarmid's inequality on  $\zeta$ , we have

$$\begin{aligned} & \Pr \left( \left| E(k(X, X')) - \frac{1}{N(N-1)} \sum_{n' \neq n} k(X_n, X_{n'}) \right| \geq \varepsilon \right) \\ & \leq \exp \left( -\frac{\varepsilon^2}{2K^2} N \right). \end{aligned} \quad (12)$$

Therefore Eq. (11) is bounded by the sum of the bound on Eq. (12) and the following:

$$\begin{aligned} & \sup_{\theta \in \Theta} \left| E(k(G_\theta(W), G_\theta(W'))) \right. \\ & \quad - \frac{1}{M(M-1)} \sum_{m \neq m'} k(G_\theta(W_m), G_\theta(W_{m'})) \\ & \quad - 2E(k(X, G_\theta(W))) \\ & \quad \left. + \frac{2}{MN} \sum_{m,n} k(X_n, G_\theta(W_m)) \right|. \end{aligned} \quad (13)$$

Thus the next step is to find the bound for the supremum above.

Define

$$\begin{aligned} & f(W_1, \dots, W_M; p_{\text{noise}}) = f(\underline{W}_M) \\ &= \sup_{\theta \in \Theta} \left| E(k(G_\theta(W), G_\theta(W'))) \right. \\ & \quad \left. - \frac{1}{M(M-1)} \sum_{m \neq m'} k(G_\theta(W_m), G_\theta(W_{m'})) \right| \end{aligned}$$

and

$$\begin{aligned} & h(X_1, \dots, X_N, W_1, \dots, W_M; p_{\text{data}}, p_{\text{noise}}) \\ &= h(\underline{X}_N, \underline{W}_M) \\ &= \sup_{\theta \in \Theta} \left| \frac{1}{MN} \sum_{m,n} k(X_n, G_\theta(W_m)) - E(k(X, G_\theta(W))) \right|. \end{aligned}$$

Then by triangle inequality, the supremum in Eq. (13) is bounded by

$$f(\underline{W}_M) + 2h(\underline{X}_N, \underline{W}_M).$$

We will first find the upper bound on  $f(\underline{W}_M)$ , i.e., for every  $\varepsilon > 0$ , we will show that there exists  $\delta_f$ , such that

$$\Pr(f(\underline{W}_M) > \varepsilon) \leq \delta_f \quad (14)$$

For each  $m \in \{1, \dots, M\}$ ,

$$\begin{aligned} & \left| f(W_1, \dots, W_m, \dots, W_M) \right. \\ & \quad \left. - f(W_1, \dots, W'_m, \dots, W_M) \right| \leq \frac{2K}{M} \end{aligned}$$

since the kernel is bounded by  $K$ , and therefore  $k(G_\theta(W_m), G_\theta(W_{m'}))$  is bounded by  $K$  for all  $m$ . The conditions of Theorem 2 are satisfied and thus we can use McDiarmid's Inequality on  $f$ :

$$\Pr(f(\underline{W}_M) - E(f(\underline{W}_M)) \geq \varepsilon) \leq \exp \left( -\frac{\varepsilon^2 M}{2K^2} \right).$$

Define

$$\mathcal{G}_k = \{k(G_\theta(\cdot), G_\theta(\cdot)) : \theta \in \Theta\}$$

To show Eq. (14), we need to bound the expectation of  $f$ . We can apply Lemma 1 on the function classes  $\mathcal{G}_k$  and  $\mathcal{G}_{k+}$ . The resulting bound is

$$E(f(\underline{W}_M)) \leq \varepsilon_{p_1} = \frac{C_f}{\sqrt{M-1}} \Psi(\gamma_1, M-1, p_1), \quad (15)$$

where  $p_1$  and  $\gamma_1$  are parameters associated with fat shattering dimension of  $\mathcal{G}_{k+}$  as stated in the assumptions of the theorem, and  $C_f$  is a constant depending on  $p_1$ .

Now we can write down the bound on  $f$ :

$$\Pr(f(\underline{W}_M) \geq \varepsilon_{p_1} + \epsilon) \leq \exp\left(-\frac{\varepsilon^2 M}{2K^2}\right) = \delta_f. \quad (16)$$

Similarly,  $h(\underline{X}_N, \underline{W}_M)$  has bounded differences:

$$\left| h(X_1, \dots, X_n, \dots, X_N, W_1, \dots, W_M) - h(X_1, \dots, X_{n'}, \dots, X_N, W_1, \dots, W_M) \right| \leq \frac{2K}{N}$$

and

$$\left| h(X_1, \dots, X_N, W_1, \dots, W_m, \dots, W_M) - h(X_1, \dots, X_N, W_1, \dots, W_{m'}, \dots, W_M) \right| \leq \frac{2K}{M}.$$

McDiarmid's inequality then implies

$$\Pr(h(\underline{X}_N, \underline{W}_M) - E(h(\underline{X}_N, \underline{W}_M)) \geq \varepsilon) \leq \exp\left(-\frac{\varepsilon^2}{2K^2} \frac{NM}{N+M}\right). \quad (17)$$

We can bound expectation of  $h(\underline{X}_N, \underline{W}_M)$  using Lemma 2 applied on  $\mathcal{G}_k^{\mathbb{X}}$  and  $\mathcal{G}_{k+}^{\mathbb{X}}$ , where

$$\mathcal{G}_k^{\mathbb{X}} = \{k(\cdot, G_\theta(\cdot)) : \theta \in \Theta\}.$$

Then

$$E(h(\underline{X}_N, \underline{W}_M)) \leq \varepsilon_{p_2} = \frac{C_h}{\sqrt{M}} \Psi(\gamma_2, M, p_2). \quad (18)$$

for some constant  $C_h$  that depends on  $p_{\otimes}$ . The final bound on  $h$  is then

$$\Pr(h(\underline{X}_N, \underline{W}_M) \geq \varepsilon_{p_2} + \varepsilon) \leq \exp\left(-\frac{\varepsilon^2}{2K^2} \frac{NM}{N+M}\right) = \delta_h.$$

Summing up the bounds from Eq. (16) and Eq. (17), it follows that

$$\Pr(f(\underline{W}_M) + 2h(\underline{X}_N, \underline{W}_M) \geq \varepsilon_{p_1} + 2\varepsilon_{p_2} + 3\varepsilon) \leq \max(\delta_f, \delta_h) = \delta_h.$$

Using the bound in Eq. (12), we have obtain the uniform bound we were looking for:

$$\Pr\left[\sup_{\theta \in \Theta} |\mathcal{E}(\theta) - \mathcal{T}(\theta)| > \varepsilon_{p_1} + 2\varepsilon_{p_2} + 4\varepsilon\right] \leq \delta_h,$$

which by Eq. (7) yields

$$\Pr\left[|\mathcal{E}(\hat{\theta}) - \mathcal{T}(\hat{\theta})| > \varepsilon_{p_1} + 2\varepsilon_{p_2} + 4\varepsilon\right] \leq \delta_h.$$

Since it was assumed that  $K = 1$  and  $N = M$ , we get  $\delta_h = \exp(-\varepsilon^2 M/4)$ .

To finish, we proceed as in the proof of Theorem 5. We can rearrange some of the terms to get a different form of Eq. (6):

$$\Pr[|\mathcal{E}(\theta^*) - \mathcal{T}(\theta^*)| > 2\varepsilon] \leq 2 \exp\left(-\frac{\varepsilon^2 M}{4}\right) = 2\delta_h.$$

All of the above implies that for any  $\varepsilon > 0$ , there exists  $\delta$ , such that

$$\Pr(\text{MMD}^2(\mathcal{H}, p_{\text{data}}, p_{\hat{\theta}}) - \text{MMD}^2(\mathcal{H}, p_{\text{data}}, p_{\theta^*}) \geq \varepsilon) \leq \delta,$$

where

$$\varepsilon = \varepsilon_{p_1} + 2\varepsilon_{p_2} + \frac{12}{\sqrt{M}} \sqrt{\log \frac{2}{\delta}}.$$

We can rewrite  $\varepsilon$  as:

$$\varepsilon = r(p_1, \gamma_1, M) + r(p_2, \gamma_2, M-1) + 12M^{-\frac{1}{2}} \sqrt{\log \frac{2}{\delta}},$$

The rate  $r(p, \gamma, N)$  is given by Eq. (15) and Eq. (18):

$$r(p, \gamma, M) = C_p \sqrt{\gamma} \begin{cases} M^{-\frac{1}{2}} & \text{if } p < 2, \\ M^{-\frac{1}{2}} \log^{\frac{3}{2}}(M) & \text{if } p = 2, \\ M^{-\frac{1}{p}} & \text{if } p > 2, \end{cases}$$

where  $C_{p_1}$  and  $C_{p_2}$  depend on  $p_1$  and  $p_2$  alone.  $\square$

We close by noting that the approximation error is zero in the nonparametric limit.

**Theorem 6** (Gretton et al. [2]). *Let  $F$  be the unit ball in a universal RKHS  $\mathcal{H}$ , defined on the compact metric space  $\mathbb{X}$ , with associated continuous kernel  $k(\cdot, \cdot)$ . Then  $\text{MMD}[\mathcal{H}, p, q] = 0$  if and only if  $p = q$ .*

**Corollary 2** (approximation error). *Assume  $p_{\text{data}}$  is in the family  $\{p_\theta\}$  and that  $\mathcal{H}$  is an RKHS induced by a characteristic kernel. Then*

$$\inf_{\theta} \text{MMD}(\mathcal{H}, p_{\text{data}}, p_\theta) = 0$$

and the infimum is achieved at  $\theta$  satisfying  $p_\theta = p_{\text{data}}$ .

*Proof.* By Theorem 6, it follows that  $\text{MMD}^2(\mathcal{H}, \cdot, \cdot)$  is a metric. The result is then immediate.  $\square$

## Acknowledgments

The authors would like to thank Bharath Sriperumbudur for technical discussions.

## References

- [1] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. “Generative Adversarial Nets”. 2014.
- [2] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola. “A Kernel Two-sample Test”. In: *J. Mach. Learn. Res.* 13 (Mar. 2012), pp. 723–773.
- [3] G. E. Hinton and R. R. Salakhutdinov. “Reducing the dimensionality of data with neural networks”. In: *Science* 313.5786 (July 2006), pp. 504–507.
- [4] O. Kallenberg. *Foundations of modern probability*. 2nd. New York: Springer, 2002, pp. xx+638.
- [5] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE*. 1998, pp. 2278–2324.
- [6] Y. Li, K. Swersky, and R. Zemel. “Generative Moment Matching Networks”. <http://arxiv.org/abs/1502.02761v1>. 2015.
- [7] D. J. MacKay. “Bayesian Neural Networks and Density Networks”. In: *Nuclear Instruments and Methods in Physics Research, A*. 1994, pp. 73–80.
- [8] S. Mendelson. “A Few Notes on Statistical Learning Theory”. English. In: *Advanced Lectures on Machine Learning*. Ed. by S. Mendelson and A. Smola. Vol. 2600. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2003, pp. 1–40.
- [9] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.
- [10] R. Salakhutdinov and G. E. Hinton. “Deep Boltzmann Machines”. In: *Journal of Machine Learning Research - Proceedings Track 5* (2009), pp. 448–455.
- [11] E. Saund. “Dimensionality-Reduction Using Connectionist Networks.” In: *IEEE Trans. Pattern Anal. Mach. Intell.* 11.3 (1989), pp. 304–314.
- [12] B. K. Sriperumbudur, A. Gretton, K. Fukumizu, G. Lanckriet, and B. Schölkopf. “Injective Hilbert Space Embeddings of Probability Measures”. In: *Conf. Comp. Learn. Theory, (COLT)*. 2008.
- [13] J. M. Susskind, A. K. Anderson, and G. E. Hinton. *The Toronto face database*. Tech. rep. 2010.
- [14] T. Tieleman and G. Hinton. *Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude*. COURSERA: Neural Networks for Machine Learning. 2012.
- [15] B. Uribe, I. Murray, and H. Larochelle. “A Deep and Tractable Density Estimator.” In: *CoRR* abs/1310.1757 (2013).

---

# Incremental Region Selection for Mini-bucket Elimination Bounds

---

**Sholeh Forouzan**

Department of Computer Science  
University of California, Irvine  
Irvine, CA, 92697

**Alexander Ihler**

Department of Computer Science  
University of California, Irvine  
Irvine, CA, 92697

## Abstract

Region choice is a key issue for many approximate inference bounds. Mini-bucket elimination avoids the space and time complexity of exact inference by using a top-down partitioning approach that mimics the construction of a junction tree and aims to minimize the number of regions subject to a bound on their size; however, these methods rarely take into account functions' values. In contrast, message passing algorithms often use "cluster pursuit" methods to select regions, a bottom-up approach in which a pre-defined set of clusters (such as triplets) is scored and incrementally added. In this work, we develop a hybrid approach that balances the advantages of both perspectives, providing larger regions chosen in an intelligent, energy-based way. Our method is applicable to bounds on a variety of inference tasks, and we demonstrate its power empirically on a broad array of problem types.

## 1 INTRODUCTION

Mini-bucket elimination (MBE) (Dechter and Rish, 2003) is a popular bounding technique for reasoning tasks defined over graphical models. MBE is often used to develop heuristic functions for search and optimization tasks (Dechter and Rish, 2003; Kask and Dechter, 2001; Marinescu and Dechter, 2007; Choi et al., 2007; Marinescu et al., 2014), although it has also been used to provide bounds on weighted counting problems such as computing the probability of evidence in Bayesian networks (Rollon and Dechter, 2010; Liu and Ihler, 2011).

MBE obtains its bounds by approximating the variable or "bucket" elimination process. Rather than exactly eliminating each variable, MBE partitions the functions into smaller sets of bounded complexity; eliminating within each partition separately gives an upper or lower bound. A sin-

gle control variable allows the user to easily trade off between accuracy and computation (memory and time). A recent extension called weighted mini-bucket (WMB) also serves to connect mini-bucket to the framework of variational bounds, allowing iterative reparameterization updates to improve the WMB bound.

The partitioning of a bucket into mini-buckets of bounded size can be accomplished in many ways, each resulting in a different accuracy. Viewed from a variational perspective, this corresponds to the critical choice of *regions* in the approximations, defining which sets of variables will be reasoned about jointly.

Traditionally, MBE is guided only by the graph structure, using a *scope-based* heuristic (Dechter and Rish, 2003) to minimize the number of buckets. However, this ignores the influence of the functions themselves on the bound. More recent extensions such as Rollon and Dechter (2010) have suggested ways of incorporating the function values into the partitioning process, with mixed success. A more bottom-up construction technique is the *relax-compensate-recover* (RCR) method of Choi and Darwiche (2010), which constructs a sequence of mini-bucket-like bounds of increasing complexity.

Variational approaches typically use a greedy, bottom-up approach termed *cluster pursuit*. Starting with the smallest possible regions, the bounds are optimized using message passing, and then new regions are added greedily from an enumerated list of clusters such as triplets (e.g., Sontag et al., 2008; Komodakis and Paragios, 2008). This technique is often very effective if only a few regions can be added, but the sheer number of regions considered often creates a computational bottleneck and prevents adopting large regions (e.g., Batra et al., 2011).

We propose a hybrid approach that is guided by the graph structure and connects to the mini-bucket construction, but takes advantage of the iterative optimization and scoring techniques of cluster pursuit. In practice, we find that our methods work significantly better than either the partitioning heuristics of Rollon and Dechter (2010), or a pure re-

gion pursuit approach. We also discuss the connections of our work to RCR (Choi and Darwiche, 2010). We validate our approach with experiments on a wide variety of problems drawn from recent UAI approximate inference competitions (Elidan et al., 2012).

## 2 PRELIMINARIES

Graphical models capture the dependencies among large numbers of random variables by explicitly representing the independence structure of the joint probability distribution. Consider a distribution

$$p(x) = \frac{1}{Z} \prod_{\alpha \in \mathcal{I}} f_{\alpha}(x_{\alpha}) \quad Z = \sum_x \prod_{\alpha} f_{\alpha}(x_{\alpha})$$

where  $x_{\alpha}$  indicates a subset of variables and  $Z$  is the normalizing constant called the *partition function*. We associate  $p(x)$  with a graph  $G = (V, E)$  where each variable  $x_i$  is associated with a node  $i \in V$  and is connected to  $x_j$  if both variables  $x_i$  and  $x_j$  are arguments of some function  $f_{\alpha}$ .  $\mathcal{I}$  is then a set of all cliques in  $G$ .

Common inference tasks include finding the most likely or MAP configuration of  $p(x)$ , a combinatorial optimization problem, or computing the partition function  $Z$ , a combinatorial summation problem. Computing  $Z$ , which corresponds to the probability of evidence in Bayesian networks, or the marginal probabilities of  $p(x)$ , are central problems in many learning and inference tasks.

### 2.1 MINI-BUCKET ELIMINATION

Unfortunately, inference tasks such as computing the partition function are often computationally intractable for many real-world problems. Exact inference, such as variable or “bucket” elimination (Dechter, 1999) is exponential in the tree-width of the model, leading to a spectrum of approximations and bounds subject to computational limits. In this section, we briefly describe bucket elimination and mini-bucket approximations.

**Bucket Elimination** (Dechter, 1999) is an exact algorithm that directly eliminates variables in sequence. Given an elimination order, BE collects all factors that include variable  $x_i$  as their earliest-eliminated argument in a *bucket*  $B_i$ , then takes their product and eliminates  $x_i$  to produce a new factor over later variables, which is placed in the bucket of its “parent”  $\pi_i$ , the earliest uneliminated variable:

$$\lambda_{i \rightarrow \pi_i}(x_{i \rightarrow \pi_i}) = \sum_{x_i} \prod_{f_{\alpha} \in B_i} f_{\alpha}(x_{\alpha}) \prod_{\lambda_{j \rightarrow i} \in B_i} \lambda_{j \rightarrow i}(x_{j \rightarrow i})$$

The functions  $\lambda_{j \rightarrow i}$  constructed during this process can be interpreted as *messages* that are passed downward in a join-tree representation of the model (Ihler et al., 2012); see Figure 1(a)-(b).

The space and time complexity of BE are exponential in the *induced width* of the graph given the elimination order. While good elimination orders can be identified using various heuristics (see e.g., Kask et al., 2011), this exponential dependence often makes direct application of BE intractable for many problems of interest.

**Minibucket Elimination.** To avoid the complexity of bucket elimination, Dechter and Rish (1997) proposed an approximation in which the factors in bucket  $B_i$  are grouped into partitions  $Q_i = \{q_i^1, \dots, q_i^p\}$ , where each partition  $q_i^j \in Q_i$ , also called a *mini-bucket*, includes no more than *ibound*+1 variables. The bounding parameter *ibound* then serves as a way to control the complexity of elimination, as the elimination operator is applied to each mini-bucket separately. Using the inequality

$$\sum_{x_i} \prod_{f_{\alpha} \in B_i} f_{\alpha} \leq \left[ \sum_{x_i} \prod_{f_{\alpha} \in q_i^1} f_{\alpha} \right] \cdot \left[ \max_{x_i} \prod_{f_{\alpha} \in q_i^2} f_{\alpha} \right],$$

MBE gives an upper bound on the true partition function, and its time and space complexity are exponential in the user-controlled *ibound*. Smaller *ibound* values result in lower computational cost, but are typically less accurate. See Figure 1(c) for an illustration.

The resulting bound depends significantly on the partitionings  $\{Q_i\}$ ; we discuss strategies for partitioning in Section 2.2.

**Weighted Mini-bucket.** A recent improvement to mini-bucket (Liu and Ihler, 2011) generalizes the MBE bound with a “weighted” elimination,

$$\sum_{x_i} \prod_{f_{\alpha} \in B_i} f_{\alpha} \leq \left[ \sum_{x_i} \prod_{f_{\alpha} \in q_i^1} f_{\alpha}^{w_1} \right]^{w_1} \cdot \left[ \sum_{x_i} \prod_{f_{\alpha} \in q_i^2} f_{\alpha}^{w_2} \right]^{w_2},$$

where  $w_i > 0$  and  $w_1 + w_2 = 1$ .

Liu and Ihler (2011) also show that the resulting bound is equivalent to a class of bounds based on tree reweighted (TRW) belief propagation (Wainwright et al., 2005), or more generally conditional entropy decompositions (CED) (Globerson and Jaakkola, 2007), on a join-graph defined by the mini-bucket procedure (see Figure 1(d)). This connection is used to derive fixed point reparameterization updates, which change the relative values of the factors  $f_{\alpha}$  while keeping their product constant in order to tighten the bound.

### 2.2 PARTITIONING METHODS

As discussed above, mini-bucket elimination and its weighted variant compute a partitioning over each bucket  $B_i$  to bound the complexity of inference and compute an upper bound on the partition function  $Z$ . However, different partitioning strategies will result in different upper

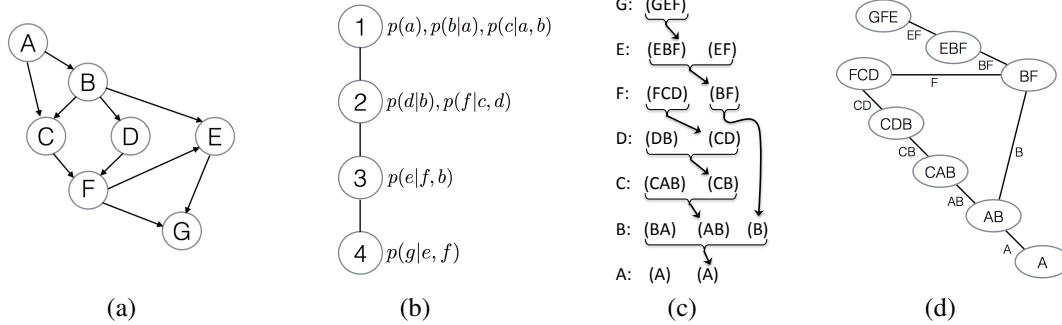


Figure 1: (a) A belief network  $P(A, B, C, D, E, F, G) = P(A) \cdot P(B|A) \cdot P(C|A, B) \cdot P(D|B) \cdot P(F|C, D) \cdot P(E|B, F) \cdot P(G|E, F)$ ; (b) a join-tree decomposition for exact inference; (c) a mini-bucket approximation ( $ibound = 2$ ), with  $F$  eliminated approximately; (d) the region or join-graph associated with (c).

bounds. Rollon and Dechter (2010) proposed a framework to study different partitioning heuristics, and compared them with the original scope based heuristic proposed by Dechter and Rish (1997). Here we summarize several approaches.

**Scope-based Partitions.** Proposed in Dechter and Rish (1997), scope-based partitioning is a top-down approach that tries to minimize the number of mini-buckets in  $B_i$  by including as many functions as possible in each mini-bucket  $q_i^k$ . To this end, it first orders the factors in  $B_i$  by decreasing number of arguments. Starting from the largest, each factor  $f_\alpha$  is then merged with the first available mini-bucket that satisfies the computational limits, i.e., where  $|\text{var}(f) \cup \text{var}(q_i^j)| \leq ibound + 1$ . If there are no mini-buckets available that can include the factor, a new mini-bucket is created and the scheme continues until all factors are assigned to a mini-bucket.

**Content-based Partitions.** Rollon and Dechter (2010), on the other hand, seeks to find a partitioning that is closest to the true bucket function,  $g_i = \sum_{X_i} \prod_{\alpha \in B_i} f_\alpha$ . This requires solving an optimization problem

$$Q^* = \arg \min_Q \text{dist}(g_i^Q, g_i)$$

where  $Q$  is a partitioning of  $B_i$  with bounding parameter  $ibound$  and

$$g_i^Q = \prod_{j=1}^p \sum_{X_i} \prod_{\alpha \in q_i^j} f_\alpha$$

is the function represented by the partitioning  $Q$ . The distance is minimized in a greedy fashion, and Rollon and Dechter (2010) studied the effectiveness of several different distance functions across multiple problem instances; however, no one distance was found to consistently outperform scope-based partitioning.

**Relax-Compensate-Recover.** Choi and Darwiche (2010) indirectly addresses the problem of partition selection within their *Relax, Compensate and Recover* framework,

in which certain equality constraints in the graph are first relaxed in order to reduce complexity of inference. New auxiliary factors are then introduced to compensate for the relaxation and enforce a weaker notion of equivalence. The recovery process then aims to identify those equivalence constraints whose relaxation were most damaging and recover them. Choi and Darwiche (2010) proposed a number of recovery heuristics, including mutual information and residual recovery.

### 2.3 VARIATIONAL BOUNDS.

The variational viewpoint of inference corresponds to optimizing an objective function over a collection of *beliefs* constrained to lie within the marginal polytope, or set of marginal probabilities that can be achieved by some joint distribution. Efficient approximations are developed by relaxing these constraints to enforce only a subset of the constraints – that the beliefs be consistent between overlapping cliques. In the case of the log partition function, we also approximate the entropy term in the objective; for example, the CED bound is:

$$\log Z \leq \max_{b_\alpha \in \mathbb{L}} \sum_{\alpha} \mathbb{E}_{b_\alpha} [\log f_\alpha] + \sum_{i,\alpha} w_{i\alpha} H(x_i | x_{\alpha \setminus i}; b_\alpha)$$

where  $\sum_{\alpha} w_{i\alpha} = 1$  for all  $i$ .

Like mini-bucket bounds, the quality of variational bounds depends significantly on the choice of regions, which determine what constraints will be enforced by the local polytope  $\mathbb{L}$  as well as the form of the entropy approximation. Traditionally, variational approximations have focused more on the optimization of the bound through message passing than the region selection aspect. Often regions are chosen to match the original model factors, and then improved using methods like cluster pursuit, described next.

**Cluster Pursuit.** Sontag et al. (2008) studied the problem of region selection for MAP inference in the context of *cluster-based* dual decomposition relaxations. They developed a bottom-up approach in which regions (typically



cycles or triplets) are added incrementally: First, the dual decomposition bound is optimized through message passing. Then, a pre-defined set of clusters, such as triplets or faces of a grid, are scored by computing a lower bound on their potential improvement of the bound; the scoring function used measures the difference between independently maximizing each pairwise factor, versus jointly maximizing over the triplet. After adding the best-scoring cluster, the procedure repeats. Similar cycle repair processes were also proposed by Komodakis and Paragios (2008) and Werner (2008), and related cluster pursuit methods have also been applied to summation problems (Welling, 2004; Hazan et al., 2012). However, scoring all possible clusters often becomes a computational bottleneck; for example, Batra et al. (2011) proposed pre-selection heuristics to reduce the number of clusters considered. In practice, cluster pursuit is usually applied to add only a few, small regions; scoring sets of larger regions is typically considered prohibitive.

### 3 A HYBRID APPROACH

Mini-bucket elimination avoids the space and time complexity of exact inference by using a top-down partitioning approach that mimics the construction of a junction tree. In contrast, message passing algorithms often use “cluster pursuit” methods to select regions, a bottom-up approach in which a predefined set of clusters (such as triplets) is scored and incrementally added.

To balance the effectiveness of both approaches, our hybrid scheme, like mini-bucket, uses the graph structure to guide region selection, while also taking advantage of the iterative optimization and scoring techniques of cluster pursuit.

Cluster pursuit algorithms use the function values, and more concretely the bound produced by them, in order to select regions that tighten the upper bound more effectively. However, there are often prohibitively many clusters to consider: for example, in a fully connected pairwise model, there are  $O(n^3)$  triplets,  $O(n^4)$  possible 4-cliques, etc., to score at each step. For this reason, cluster pursuit methods typically restrict their search to a predefined set of clusters, such as triplets Sontag et al. (2008). Our proposed approach uses the graph structure to guide the search for regions, restricting the search to merges of existing clusters, within one bucket at a time. This allows us to restrain the complexity of the search and add larger regions more effectively.

In contrast, the content-based heuristics for region selection of Rollon and Dechter (2010) use the graph structure as a guide, but their scoring scheme only takes into account the messages from the earlier buckets in the elimination order. Our proposed hybrid approach uses iterative optimization on the junction tree in order to make more effective partitioning decisions.

---

#### Algorithm 1 Incremental region selection for WMBE

---

**Input:** factor graph  $(G)$ , bounding parameter  $ibound$  and maximum number of iterations  $T$   
**Initialize**  $wmb$  to a join graph using e.g. a min-fill ordering  $o$ , uniform weights and uniform messages  
**for** each bucket  $B_i$  following the elimination order **do**  
    **repeat**  
         $(q_i^m, q_i^n) \leftarrow \text{SelectMerge}(Q_i)$   
         $\mathcal{R} \leftarrow \text{AddRegions}(wmb, o, q_i^m, q_i^n)$   
         $wmb \leftarrow \text{MergeRegions}(wmb, \mathcal{R})$   
    **for**  $iter = 1$  to  $T$  **do**  
        // pass forward messages and reparameterize:  
         $wmb \leftarrow \text{msgForward}(wmb)$   
        // pass backward messages:  
         $wmb \leftarrow \text{msgBackward}(wmb)$   
    **end for**  
**until** no more merges possible  
**end for**

---

Algorithm 1 describes the overall scheme of our hybrid approach, which is explained in detail next.

#### 3.1 INITIALIZING A JOIN TREE

Given a factor graph  $G$  and a bounding parameter  $ibound$ , we start by initializing a join graph, using a min-fill elimination ordering (Dechter, 2003)  $\mathbf{o} = \{x_1, \dots, x_n\}$  and  $ibound = 1$ . For any given bucket  $B_i$ , this results in each mini-bucket  $k \in B_i$  containing a single factor  $f_\alpha$ . We denote the result of the elimination as  $\lambda_{k \rightarrow l}$  which is sent to the bucket  $B_j$  of its first-eliminated argument in  $\mathbf{o}$ . Here,  $l = \text{pa}(k)$  denotes the parent region of  $k$  which can be one of the initial mini-buckets in  $B_j$  if  $\text{var}(\lambda_{k \rightarrow l}) \subseteq \text{var}(l)$ , or be a new mini-bucket with  $f_l = 1$ . In our implementation we choose  $l \in B_j$  to be the mini-bucket with the largest number of arguments,  $|\text{var}(l)|$ , such that  $\text{var}(\lambda_{k \rightarrow l}) \subseteq \text{var}(l)$ .

Using weighted mini-bucket for our elimination scheme, we initialize the mini-bucket weights  $w_r$  uniformly within each bucket  $B_i$ , so that for  $r \in Q_i$ ,  $w_r = \frac{1}{|Q_i|}$ , which ensures  $\sum_{r \in Q_i} w_r = 1$ .

#### 3.2 MESSAGE PASSING

We use iterative message passing on the join graph to guide the region selection decision. Having built an initial join graph, we use the weighted mini-bucket messages (Liu and Ihler, 2011) to compute forward and backward messages, and perform reparameterization of the functions  $f_\alpha$  to tighten the bound.

Let  $r$  be a region of the mini-bucket join graph, and  $s$  its parent,  $s = \text{pa}(r)$ , with weights  $w_r$  and  $w_s$ , and  $f_r(x_r)$  the product of factors assigned to region  $r$ . Then we compute

the forward messages as,

**Forward Messages:**

$$\lambda_{r \rightarrow s}(x_s) = \left[ \sum_{x_r \setminus x_s} [f_r(x_r) \prod_{t:s=\text{pa}(t)} \lambda_{t \rightarrow s}(x_s)]^{\frac{1}{w_s}} \right]^{w_s} \quad (1)$$

and compute the upper bound using the product of forward messages computed at roots of the join graph,

**Upper bound on Z:**

$$Z \leq \prod_{r:\text{pa}(r)=\emptyset} \lambda_{r \rightarrow \emptyset} \quad (2)$$

In order to tighten the bound, we compute backward messages in the join graph,

**Backward Messages:**

$$\lambda_{s \rightarrow r}(x_r) = \left[ \sum_{x_s \setminus x_r} [f_s(\cdot) \prod_t \lambda_{t \rightarrow s}(\cdot)]^{\frac{1}{w_s}} [\lambda_{r \rightarrow s}(\cdot)]^{-\frac{1}{w_r}} \right]^{w_r}$$

where  $t$  indexes all neighbors (parent and children) of region  $s$ . We then use these incoming messages to compute a weighted belief at region  $r$ , and reparameterize the factors  $f_r$  for each region  $r$  in a given bucket  $B_i$  (e.g.,  $r \in Q_i$ ) to enforce a weighted moment matching condition:

**Reparameterization:**

$$\begin{aligned} b_r(x_i) &= \sum_{x_r \setminus x_i} [f_r(x_r) \prod_t \lambda_{t \rightarrow r}(x_r)]^{\frac{1}{w_r}} \\ \bar{b}(x_i) &= \left[ \prod_{r \in Q_i} b_r(x_i) \right]^{1/\sum_r w_r} \\ f_r(x_r) &\leftarrow f_r(x_r) [\bar{b}(x_i)/b_r(x_i)]^{w_r} \end{aligned}$$

In practice, we usually match on the variables present in all mini-buckets  $r$ , e.g.,  $\cap_{r \in Q_i} x_r$ , rather than just  $x_i$ ; this gives a tighter bound for the same amount of computation.

**3.3 ADDING NEW REGIONS**

New regions are added to the initial join tree after one or more rounds of iterative optimization. To contain the complexity of the search over clusters, we restrict our attention to pairs of mini-buckets to merge within each bucket. To do so, we also use the elimination order  $o$  to guide our search, processing each bucket  $B_i$  one at a time in order.

Given bucket  $B_i$  and current partitioning  $Q_i = \{q_i^1, \dots, q_i^k\}$ , we score the merge for each allowed pair of mini-buckets  $(q_i^m, q_i^n)$ , e.g., those with  $|\text{var}(q_i^m) \cup \text{var}(q_i^n)| \leq \text{ibound} + 1$ , using an estimate of the benefit to the bound that may arise from merging the pair:

$$S(q_i^m, q_i^n) = \max_x [\lambda_{m \rightarrow \pi_m}(x_{\pi_m}) \times \lambda_{n \rightarrow \pi_n}(x_{\pi_n}) \div \lambda_{r \rightarrow \pi_r}(x_r)]$$

This score can be justified as a lower bound on the decrease in  $\log Z$ , since it corresponds to adding region  $\pi_r$  with weight  $w_{\pi_r} = 0$ , while reparameterizing the parents  $\pi_m, \pi_n$  to preserve their previous beliefs. This procedure leaves the bound unchanged except for the contribution of  $\pi_r$ ; eliminating with  $w_{\pi_r} = 0$  is equivalent to the max operation. For convenience, we set  $S(q_i^m, q_i^n) = 0$  for pairs which violate the *ibound* constraint. Then, having computed the score between all pairs, we choose the pair with maximum score to be merged into a new clique. In Algorithm 1, the function `SelectMerge( $\cdot$ )` denotes this scoring and selection process.

**3.4 UPDATING GRAPH STRUCTURE**

Having found which mini-buckets to merge, we update the join graph to include the new clique  $r = q_i^m \cup q_i^n$ . Our goal is to add the new region such that it affects the scope of the existing regions in the join tree as little as possible. Adding the new clique is done in two steps:

First we initialize a new mini-bucket in  $B_i$  with its scope matching  $\text{var}(r)$ . Eliminating variable  $x_i$  from this new mini-bucket results in the message  $\lambda_{r \rightarrow \pi_r}$ . The earliest argument of  $\lambda_{r \rightarrow \pi_r}$  in the elimination order determines the bucket  $B_j$  containing mini-buckets that can potentially be the parent,  $\pi_r$ , of the new region. To find  $\pi_r$  in  $B_j$  we seek a mini-bucket  $q_j^k$  that can contain  $r$ , i.e.,  $\text{var}(\lambda_{r \rightarrow \pi_r}) \subseteq \text{var}(q_j^k)$ . If such a mini-bucket exists, we set  $\pi_r$  to  $q_j^k$ ; otherwise, we create a new mini-bucket  $q_j^{|Q_j|+1}$  and add it to  $Q_j$ , with a scope that matches  $\text{var}(\lambda_{r \rightarrow \pi_r})$ . The same procedure is repeated after eliminating  $x_j$  from  $q_j^{|Q_j|+1}$  until we either find a mini-bucket already in the join tree that can serve as the parent, or  $\text{var}(\lambda_{r \rightarrow \pi_r}) = \emptyset$  in which case the newly added mini-bucket is a root. Algorithm 2 describes these initial structural modifications.

Having added the new regions, we then try to remove any unnecessary mini-buckets, and update both the join tree and the function values of the newly added regions to ensure that the bound is improved. To this end, we update every new mini-bucket  $r$  that was added to the join tree in the previous step as follows. For mini-bucket  $r \in Q_i$ , we first find any mini-buckets  $s \in Q_i$  that can be subsumed by  $r$ , i.e.,  $\text{var}(s) \subseteq \text{var}(r)$ . For each of these mini-buckets  $s$ , we connect all of  $s$ 's children (mini-buckets  $t$  such that  $\text{pa}(t) = s$ ) to  $r$ , e.g., set  $\text{pa}(t) = r$ . We also merge the factors associated with  $r$  and  $s$ , so that  $f_r \leftarrow f_r \times f_s$ .

Next, we reparameterize several other functions in the join graph in order to preserve or improve the current bound

---

**Algorithm 2** AddRegions: find regions to add for merge

---

**Input:** The join graph  $wmb$ , elimination order  $o$ , and mini-buckets  $q_i^m$  and  $q_i^n$  to be merged

**Output:** a list of newly added mini-buckets  $\mathcal{R}$

**Initialize** new region  $q^r$  with  $\text{var}(q^r) = \text{var}(q_i^m \cup q_i^n)$  and add it to  $Q_i$

**repeat**

**Update**  $\mathcal{R} = \mathcal{R} \cup q^r$

**Set** new clique  $C = \text{var}(q^r) \setminus x_i$

**if**  $C = \emptyset$  **then**

$done \leftarrow True$

**else**

**Find**  $B_j$  corresponding to the first un-eliminated variable in  $C$  based on elimination order  $o$

**for** each mini-bucket region  $q_j^k \in Q_j$  **do**

**if**  $C \subseteq \text{var}(q_j^k)$  **then**

        //forward message fits in existing mini-bucket:

$done \leftarrow True$

**end if**

**end for**

**end if**

**if** not  $done$  **then**

    // Create a new region to contain forward message:

**Initialize** new region  $q^r$  with  $\text{var}(q^r) = C$  and add it to  $Q_j$

**end if**

**until** done

---

value. Specifically, removing  $s$  changes the incoming, forward messages to its parent,  $\pi_s = \text{pa}(s)$ , which changes the bound. By reparameterizing the factor at  $\pi_s$ ,

$$f_{\pi_s} \leftarrow f_{\pi_s} \times \lambda_{s \rightarrow \pi_s} \quad f_{\pi_r} \leftarrow f_{\pi_r} \div \lambda_{s \rightarrow \pi_s}$$

we keep the overall distribution unchanged, but ensure that the bound is strictly decreased.

Finally we remove  $s$  from  $Q_i$ , completing the merge of mini-buckets  $s$  and  $r$ . This process is given in Algorithm 3 and depicted in Figure 2 for a small portion of join-graph.

Every merge decision is followed by one or more iterations of message passing, followed by rescaling the mini-buckets in  $B_i$ . The process of message passing and merging continues until no more mini-buckets of  $B_i$  can be merged, while satisfying the bounding parameter  $ibound$ .

Continuing along the elimination order, the same procedure is repeated for the mini-buckets in each bucket  $B_i$ , and the final upper bound to the partition function is computed using Eq. (2).

## 4 DISCUSSION

Our method is similar to context-based mini-buckets, with the main difference being that message passing performed

---

**Algorithm 3** MergeRegions: merge and parameterize newly added regions to improve bound

---

**Input:** The join graph  $wmb$  and a list of newly added mini-buckets  $\mathcal{R}$

**for all**  $r \in \mathcal{R}$  **do**

**Initialize** new region  $r$  in  $B_i$  with  $f_r(x_r) = 1$

**Find** regions  $\{s \mid s \in Q_i \ \& \ \text{var}(s) \subseteq \text{var}(r)\}$

  // Remove / merge contained regions  $s$ :

**for all** found regions  $s$  **do**

    Connect all children of  $s$  to  $r$

$f_r = f_r \cdot f_s$  // merge factors and

    // preserve belief at parent  $\pi_s$ :

$f_{\pi_s} = f_{\pi_s} \times \lambda_{s \rightarrow \pi_s}$

$f_{\pi_r} = f_{\pi_r} \div \lambda_{s \rightarrow \pi_s}$

    Remove  $s$  from  $Q_i$

**end for**

**end for**

---

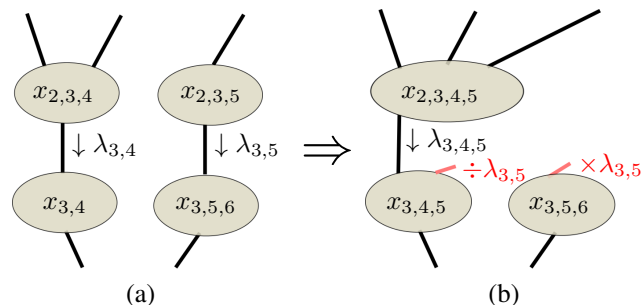


Figure 2: Merge and post-merge reparameterization operations. (a) A portion of a join-graph corresponding to the elimination of  $x_2$  and  $x_3$ , each with two mini-buckets. (b) Merging cliques  $(2, 3, 4)$  and  $(2, 3, 5)$  produces a new clique  $(3, 4, 5)$ , which subsumes and removes clique  $(3, 4)$ . Having removed parent  $(2, 3, 5)$ , we reparameterize the new clique functions by the original message  $\lambda_{3,5}$  (red) to preserve the original belief at  $(3, 5, 6)$  and ensure that the bound is tightened. See text for more detail.

on the simpler graph is used to reparameterize the functions before the merge scores are computed.

Our method can also be viewed as a cluster pursuit approach, in which we restrict the clusters considered, to unions of the current minibuckets at the earliest bucket  $B_i$ , and merge up to our computational limit before moving on to later buckets. These restrictions serve to reduce the number of clusters considered, but in addition, appear to lead to better regions than a purely greedy region choice – in the experiments (Section 5), we compare our approach to a more “cluster pursuit-like” method, in which pairs of regions in *any* bucket  $B_i$  are considered and scored. Perhaps surprisingly, we find that this greedy approach actually gives significantly worse regions overall, suggesting

that processing the buckets in order can help by avoiding creating unnecessary regions.

Finally, our method is also closely related to RCR (Choi and Darwiche, 2010). From this perspective, we “relax” to a low-*ibound* minibucket, “compensate” by variational message passing, and “recover” by selecting regions that will tighten the variational bound defined by the join graph. Compared to RCR, we find a number of differences in our approach: (1) RCR selects constraints to recover anywhere in the graph, similar to a greedy cluster pursuit; as noted, this appears to work significantly less well than an ordered recovery process. (2) RCR makes its recovery updates to the relaxed graph, then (re)builds a (new) join tree over the relaxed graph; in contrast, we incrementally alter the join graph directly, which avoids starting from scratch after each merge. (3) Our method is solidly grounded in the theory of variational bounds and message passing, ensuring that both the message passing and region merging steps are explicitly tightening the same bound. From this perspective, for example, it becomes clear that RCR’s “residual recovery” heuristic is unlikely to be effective, since after message passing, the reparameterization updates should ensure that all mini-buckets containing a variable  $x_i$  will match on their marginal beliefs. In other words, residual recovery is making its structure (region) choices using a criterion that actually measures mismatches that can be resolved by message passing.

## 5 EMPIRICAL EVALUATION

To show our method’s effectiveness compared to previous region selection strategies for MBE, we tested our incremental approach on a number of real world problems drawn from past UAI approximate inference challenges, including linkage analysis, protein side chain prediction, and segmentation problems. We compare our hybrid region selection method against the scope-based heuristic of Dechter and Rish (1997) and the content-based heuristic of Rollon and Dechter (2010).

**Experimental Setup.** For each set of experiments, we initialize a join tree using WMB elimination with  $ibound = 1$ . We use an elimination ordering found using the min-fill heuristic (Dechter, 2003) and set the weights uniformly in each bucket. As a result, each mini-bucket  $q_i^k$  contains a single factor  $f_\alpha$  as described in section 3.1.

From this initial setup, we then use Algorithm 1 to merge mini-buckets incrementally and compute the upper bound as in Eq. (2).

**Segmentation.** To evaluate the different methods on pairwise binary problems we used a set of segmentation models from the UAI08 approximate inference challenge. These models have  $\approx 230$  binary variables and  $\approx 850$  factors. We

used varying *ibounds* for comparison and report the results on two values,  $ibound \in [5, 10]$ . Table 1 compares the upper bound on the log partition function for a representative subset of instances in this category, for two different computational limits,  $ibound = 5$  and  $ibound = 10$ . Different columns show the bound achieved using different partitioning heuristics:

- (1) **Sep** represents naïve scope-based partitioning;
- (2) **Cont** represents the energy based heuristic of Rollon and Dechter (2010); and
- (3) **Hyb** represents our hybrid approach, interleaving iterative optimization with partitioning.

The results show clear improvement in the upper bound using our hybrid approach, indicating the effectiveness of iterative message passing and optimization in guiding region selection. To further study the effectiveness of the merged regions in the context of message passing and optimization, we then fully optimized the join-graphs generated by the three region selection schemes using iterative message passing until convergence. The upper bounds after such optimization are denoted by *inst-opt* for each problem instance, *inst*. As might be expected, this additional optimization step improves the bounds of the scope-based and content-based heuristics more dramatically than our hybrid method; however, even after full optimization of the bounds, we find that the hybrid method’s bounds remain better in all of the 6 instances except one, indicating that our method has identified fundamentally better regions than the previous approaches.

**Linkage Analysis.** To compare the various methods on models with non-pairwise factors and higher cardinalities of variables, we studied pedigree models. The pedigree linkage analysis models from the UAI08 approximate inference challenge have  $\approx 300 - 1000$  variables, whose cardinalities vary in the range of  $[2, \dots, 5]$ ; the induced width of the models are typically  $\approx 20 - 30$ . We used varying *ibounds* for comparison and report the results on two values  $ibound \in [5, 10]$ .

Table 2 shows the upper bounds on a subset of pedigree problems, again showing the effectiveness of the hybrid method: we find that again, the hybrid method consistently outperforms the other two region selection approaches, and results in better fully optimized bounds in all of the 22 instances when  $ibound = 5$  and all but two cases when  $ibound = 10$ .

**Effect of *ibound*.** We further studied the results of the three partitioning methods across a range of *ibounds* to compare the effectiveness of our method when *ibound* is set to a range of values. Figure 3 shows the results for an instance of pedigree dataset. As shown here, our method is more effective on smaller *ibounds*, where there are a large number of possible merges and finding the best one results

Table 1: **UAI Segmentation Instances.** Different columns show the bound achieved using each partitioning heuristic, where “Scp”, “Cont” and “Hyb” represent the naïve scope based partitioning for MBE (Dechter and Rish, 1997), the context (or energy) based heuristic of Rollon and Dechter (2010) and our hybrid approach interleaving iterative optimization with partitioning, respectively. In all but one case, our proposed construction provides tighter bounds.

| Instance   | <i>ibound = 5</i> |          |                 | <i>ibound = 10</i> |          |                 |
|------------|-------------------|----------|-----------------|--------------------|----------|-----------------|
|            | Scp               | Cont     | Hyb             | Scp                | Ctxt     | Hyb             |
| 2-17-s     | -31.3197          | -33.4840 | <b>-49.5670</b> | -38.9801           | -42.1524 | <b>-52.507</b>  |
| 2-17-s-opt | -46.9314          | -45.4286 | <b>-49.6432</b> | -48.661            | -48.4306 | <b>-52.5633</b> |
| 8-18-s     | -54.9884          | -60.9899 | <b>-85.6518</b> | -72.3045           | -72.284  | <b>-87.2385</b> |
| 8-18-s-opt | -80.6527          | -81.1391 | <b>-85.6694</b> | -83.0398           | -79.0921 | <b>-87.2556</b> |
| 9-24-s     | -51.2897          | -49.3903 | <b>-55.6046</b> | -54.9325           | -55.0151 | <b>-55.615</b>  |
| 9-24-s-opt | <b>-56.0513</b>   | -53.7852 | -55.6241        | -54.9325           | -55.0151 | <b>-55.615</b>  |
| 17-4-s     | -58.7953          | -59.0758 | <b>-81.5415</b> | -80.267            | -79.3323 | <b>-85.3712</b> |
| 17-4-s-opt | -71.7959          | -76.8213 | <b>-81.6079</b> | -83.2573           | -82.9646 | <b>-85.3865</b> |
| 7-11-s     | -59.8250          | -57.2773 | <b>-72.7178</b> | -71.1296           | -70.1542 | <b>-75.2869</b> |
| 7-11-s-opt | -70.7037          | -68.8255 | <b>-72.9556</b> | -74.6424           | -73.9855 | <b>-75.2905</b> |

Table 2: **UAI Pedigree Instances.** Different columns show the bound achieved using each partitioning heuristic; again, “Scp”, “Cont” and “Hyb” are scope based partitioning (Dechter and Rish, 1997), the context-based heuristic (Rollon and Dechter, 2010) and our proposed, hybrid approach. In all cases, our proposed construction provides stronger bounds, both before and after full optimization using message passing.

| Instance  | <i>ibound = 5</i> |          |                 | <i>ibound = 10</i> |           |                  |
|-----------|-------------------|----------|-----------------|--------------------|-----------|------------------|
|           | Scp               | Cont     | Hyb             | Scp                | Ctxt      | Hyb              |
| ped23     | -67.8848          | -69.9015 | <b>-71.9677</b> | -75.6057           | -78.4033  | <b>-79.4649</b>  |
| ped23-opt | -71.6951          | -71.6988 | <b>-72.0670</b> | -76.0531           | -78.7646  | <b>-79.4669</b>  |
| ped20     | -35.6986          | -40.1787 | <b>-44.7230</b> | -51.2648           | -54.4136  | <b>-57.6506</b>  |
| ped20-opt | -42.3024          | -42.8980 | <b>-44.7501</b> | -52.6043           | -56.2193  | <b>-57.7841</b>  |
| ped42     | -41.6656          | -43.5206 | <b>-51.0000</b> | -55.0681           | -57.5755  | <b>-61.3504</b>  |
| ped42-opt | -49.0089          | -50.0585 | <b>-51.1018</b> | -57.3718           | -59.2170  | <b>-61.3560</b>  |
| ped38     | -79.4742          | -89.6906 | <b>-92.7643</b> | -98.6339           | -101.1178 | <b>-113.6004</b> |
| ped38-opt | -83.0351          | -91.8510 | <b>-93.0615</b> | -101.0715          | -104.1031 | <b>-113.8926</b> |
| ped19     | -58.9234          | -63.2737 | <b>-80.6488</b> | -90.7840           | -93.9027  | <b>-100.3230</b> |
| ped19-opt | -72.3311          | -77.7023 | <b>-80.7167</b> | -92.8916           | -96.2846  | <b>-100.3388</b> |

in a greater improvement to the upper bound. For larger *ibound*s, the upper bounds produced by all three heuristics are fairly close.

**Protein Side-Chain Prediction.** Finally, to examine models over high-cardinality variables, we look at a subset of the protein side chain prediction models, originally from Yanover and Weiss (2003) and Yanover et al. (2006). These models contain  $\approx 300 - 1000$  variables with cardinalities between 2 and 81, with pairwise potential functions. For these problems, we only ran our experiments

using *ibound = 2*, due to the high number of states for each variable. Table 3 shows the results of the three partitioning methods, which again agrees with the previous experiments: our hybrid method outperforms the other two in all 44 instances in this problem set, both before and after the bound is fully optimized.

**Greedy vs. Elimination Order Based Merging.** As discussed before, we restrict the clusters considered for merges to unions of the current minibuckets at the earliest bucket  $B_i$ , and merge up to our computational limit be-

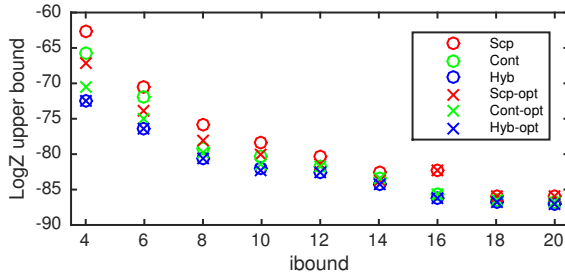


Figure 3: The upper bound achieved by the three partitioning heuristics for pedigree23 instance over  $ibound$  range between 4 to 20.

Table 3: **Protein side-chain prediction.** Here we show results for only  $ibound = 2$ , due to the high number of states in each variable. Our method often produces dramatically better partitionings than scope- or context-based mini-bucket partitions.

| Instance  | Scp       | Cont     | Hyb             |
|-----------|-----------|----------|-----------------|
| 1crz      | -242.2036 | -284.865 | <b>-451.598</b> |
| 1crz -opt | -528.5348 | -495.514 | <b>-545.929</b> |
| 2cav      | 71.2802   | -26.0052 | <b>-148.637</b> |
| 2cav -opt | -156.5387 | -240.289 | <b>-272.606</b> |
| 1kk1      | 89.5527   | 46.8216  | <b>-121.723</b> |
| 1kk1 -opt | -115.4737 | -105.894 | <b>-143.447</b> |
| 1e4f      | 40.6686   | -6.1785  | <b>-190.943</b> |
| 1e4f -opt | -212.4607 | -202.547 | <b>-240.27</b>  |
| 1ehg      | 71.3308   | 14.768   | <b>-149.158</b> |
| 1ehg -opt | -169.8435 | -147.333 | <b>-211.678</b> |

fore moving on to later buckets, which serves to reduce the number of clusters considered. We compare our choice of clusters with a purely greedy region choice in which pairs of regions in *any* bucket  $B_i$  are considered and scored.

Interestingly the upper bounds achieved using the greedy approach was not better than the top down merging based on elimination order. The reason for this behavior is that the top-down approach allows large regions generated by mini-buckets early in the elimination ordering to be processed by buckets later in the order; the greedy approach disrupts this flow and results in extra regions that cannot be merged with any other region while respecting the  $ibound$ .

## 6 CONCLUSION

We presented a new merging heuristic for (weighted) mini-bucket elimination that uses message passing optimization of the bound, and variational interpretations, in order to construct a better heuristic for selecting moderate to large regions in an intelligent, energy-based way. Our approach inherits the advantages of both cluster pursuit in variational

Table 4: **Top-down vs. Greedy Merging.** We examine the effect of using a “fully greedy” merging procedure closer to standard cluster pursuit, in which we merge the best-scoring cluster in any bucket at each step. We find that following the top-down ordering actually results in significantly better bounds. Results shown are for  $ibound = 5$ .

| Instance  | Top-Down        | Greedy   |
|-----------|-----------------|----------|
| ped23     | <b>-71.9677</b> | -67.9094 |
| ped23-opt | <b>-72.0670</b> | -67.9094 |
| ped20     | <b>-44.7230</b> | -38.0717 |
| ped20-opt | <b>-44.7501</b> | -38.0718 |
| ped42     | <b>-49.9955</b> | -37.8576 |
| ped42-opt | <b>-50.0469</b> | -37.8582 |
| ped38     | <b>-92.7643</b> | -79.9144 |
| ped38-opt | <b>-93.0615</b> | -79.9144 |
| ped19     | <b>-80.6488</b> | -48.6900 |
| ped19-opt | <b>-80.7167</b> | -48.6904 |

inference, and (weighted) mini-bucket elimination perspectives to produce a tight bound. We validated our approach with experiments on a wide variety of problems drawn from a recent UAI approximate inference competition. In practice, we find that our methods work significantly better than either existing partitioning heuristics for mini-bucket (Rollon and Dechter, 2010), or a pure region pursuit approach. We expect this construction to improve our ability to search and solve large problems. However, our method does involve additional computational overhead compared to, say, scope-based constructions, in order to evaluate and make merge decisions. We did not focus here on any-time performance; a more nuanced balance of time, memory, and bound quality is one direction of potential future study.

## Acknowledgements

This work is supported in part by NSF grants IIS-1065618 and IIS-1254071, and by the United States Air Force under Contract No. FA8750-14-C-0011 under the DARPA PPAML program.

## References

- D. Batra, S. Nowozin, and P. Kohli. Tighter relaxations for map-mrf inference: A local primal-dual gap based separation algorithm. *JMLR - Proceedings Track*, 15: 146–154, 2011.
- Arthur Choi and Adnan Darwiche. Relax, compensate and then recover. In *New Frontiers in Artificial Intelligence - JSAI-isAI 2010 Workshops, LENLS, JURISIN, AMBN*,

- ISS, Tokyo, Japan, November 18-19, 2010, Revised Selected Papers*, pages 167–180, 2010.
- Arthur Choi, Mark Chavira, and Adnan Darwiche. Node splitting: A scheme for generating upper bounds in bayesian networks. In *UAI*, pages 57–66. AUA Press, 2007.
- R. Dechter and I. Rish. Mini-buckets: A general scheme of approximating inference. *Journal of ACM*, 50(2):107–153, 2003.
- Rina Dechter. Bucket elimination: A unifying framework for reasoning. *Artificial Intelligence*, 113(12):41 – 85, 1999.
- Rina Dechter. *Constraint Processing*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003. ISBN 1558608907.
- Rina Dechter and Irina Rish. A scheme for approximating probabilistic inference. In *Proc. Uncertainty in Artificial Intelligence (UAI)*, pages 132–141, 1997.
- G. Elidan, A. Globerson, and U. Heinemann. PASCAL 2011 probabilistic inference challenge. <http://www.cs.huji.ac.il/project/PASCAL/>, 2012.
- A. Globerson and T.S. Jaakkola. Approximate inference using conditional entropy decompositions. In *In Proceedings of the 11th International Conference on Artificial Intelligence and Statistics (AISTATS-07)*, 2007.
- T. Hazan, J. Peng, and A. Shashua. Tightening fractional covering upper bounds on the partition function for high-order region graphs. In *Uncertainty in Artificial Intelligence*, 2012.
- Alexander Ihler, Natalia Flerova, Rina Dechter, and Lars Otten. Join-graph based cost-shifting schemes. In *Uncertainty in Artificial Intelligence (UAI)*, pages 397–406. AUA Press, Corvallis, Oregon, August 2012.
- K. Kask and R. Dechter. A general scheme for automatic generation of search heuristics from specification dependencies. *Artificial Intelligence*, 129(1-2):91–131, 2001.
- Kalev Kask, Andrew Gelfand, Lars Otten, and Rina Dechter. Pushing the power of stochastic greedy ordering schemes for inference in graphical models. In *AAAI’11*, pages –1–1, 2011.
- N. Komodakis and N. Paragios. Beyond loose LP-relaxations: Optimizing MRFs by repairing cycles. pages 806–820, 2008.
- Qiang Liu and Alexander Ihler. Bounding the partition function using hölder’s inequality. In Lise Getoor and Tobias Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, ICML ’11, pages 849–856, New York, NY, USA, June 2011. ACM. ISBN 978-1-4503-0619-5.
- R. Marinescu and R. Dechter. Best-first and/or search for most probable explanations. In *Uncertainty in Artificial Intelligence (UAI)*, 2007.
- R. Marinescu, R. Dechter, and A. Ihler. AND/OR search for marginal MAP. In *International Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 563–572, 2014.
- Emma Rollon and Rina Dechter. Evaluating partition strategies for mini-bucket elimination. In *International Symposium on Artificial Intelligence and Mathematics (ISAIM 2010)*, Fort Lauderdale, Florida, USA, January 6-8, 2010, 2010.
- D. Sontag, T. Meltzer, A. Globerson, T. Jaakkola, and Y. Weiss. Tightening lp relaxations for map using message passing. In *Uncertainty in Artificial Intelligence*, pages 503–510, 2008.
- M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. A new class of upper bounds on the log partition function. 51 (7):2313–2335, July 2005.
- M. Welling. On the choice of regions for generalized belief propagation. In *Uncertainty in Artificial Intelligence*, pages 585–592, 2004.
- T. Werner. High-arity interactions, polyhedral relaxations, and cutting plane algorithm for soft constraint optimization (map-mrf). In *Computer Vision and Pattern Recognition*, 2008.
- Chen Yanover and Yair Weiss. Approximate inference and protein-folding. In S. Thrun S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 1457–1464. MIT Press, Cambridge, MA, 2003.
- Chen Yanover, Talya Meltzer, and Yair Weiss. Linear programming relaxations and belief propagation - an empirical study. *Journal of Machine Learning Research*, 7: 1887–1907, 2006.

---

# Estimating Mutual Information by Local Gaussian Approximation

---

**Shuyang Gao**

Information Sciences Institute  
University of Southern California  
sgao@isi.edu

**Greg Ver Steeg**

Information Sciences Institute  
University of Southern California  
gregv@isi.edu

**Aram Galstyan**

Information Sciences Institute  
University of Southern California  
galstyan@isi.edu

## Abstract

Estimating mutual information (MI) from samples is a fundamental problem in statistics, machine learning, and data analysis. Recently it was shown that a popular class of non-parametric MI estimators perform very poorly for strongly dependent variables and have sample complexity that scales exponentially with the true MI. This undesired behavior was attributed to the reliance of those estimators on local uniformity of the underlying (and unknown) probability density function. Here we present a novel semi-parametric estimator of mutual information, where at each sample point, densities are *locally* approximated by a Gaussians distribution. We demonstrate that the estimator is asymptotically unbiased. We also show that the proposed estimator has a superior performance compared to several baselines, and is able to accurately measure relationship strengths over many orders of magnitude.

## 1 Introduction

Mutual information (MI) is a fundamental measure of dependence between two random variables. While it initially arose in the theory of communication as a natural measure of ability to communicate over noisy channels (Shannon, 1948), mutual information has since been used in different disciplines such as machine learning, information retrieval, neuroscience, and computational biology, to name a few. This widespread use is due in part to the generality of the measure, which allows it to characterize dependency strength for both linear and non-linear relationships between arbitrary random variables.

Let us consider the following basic problem, where, given a set of i.i.d. samples from an unknown, absolutely continuous joint distribution, our goal is to estimate the mutual information from these samples. A naive method would

be first to learn the underlying probability distribution using either parametric or non-parametric methods, and then calculate the mutual information from the obtained distribution. Unfortunately, this naive approach often fails, as it requires a very large number of samples, especially in high dimensions. A different approach is to estimate mutual information directly from samples. For instance, rather than estimating the whole probability distribution, one could estimate the density (and its marginals) only at each sample point, and then plug those estimates into the expression for mutual information. This type of direct estimators been shown to be a more feasible method for estimating MI in higher dimensions. An important and very popular class of such estimators is based on k-nearest-neighbor (kNN) graphs and their generalizations (Singh et al., 2003; Kraskov et al., 2004; Pál et al., 2010).

Despite the widespread popularity of the direct estimators, it was recently demonstrated that those methods fail to accurately estimate mutual information for *strongly dependent* variables (Gao et al., 2015). Specifically, it was shown that accurate estimation of mutual information between two strongly dependent variables requires a number of samples that scales *exponentially* with the true mutual information. This undesired behavior was contributed to the assumption of local uniformity of the underlying distribution postulated by those estimators. To address this shortcoming, (Gao et al., 2015) proposed to add a correction term to compensate for non-uniformity, based on local PCA-induced neighborhoods. Although intuitive, the resulting estimator relied on a heuristically tuned threshold parameter and had no theoretical performance guarantees (Gao et al., 2015).

Our main contribution is to propose a novel mutual information estimator based on *local Gaussian approximation*, with provable performance guarantees, and superior empirical performance compared to existing estimators over a wide range of relationship strength. Instead of assuming a uniform distribution in the local neighborhood, our new estimator assumes a Gaussian distribution *locally* around each point. The new estimator leverages previous results on *local likelihood density estimation* (Hjort and Jones, 1996;



Loader, 1996). As our main theoretical result, we demonstrate that the new estimator is asymptotically unbiased. We also demonstrate that the proposed estimator performs as well as existing baseline estimators for weak relationships, but outperforms all of those estimators for stronger relationships.

The paper is organized as follows. In the next section, we review the basic definitions of information-theoretic concepts such as mutual information and formally define our problem. In section 3, we review the limitations of current mutual information estimators as pointed out in (Gao et al., 2015). Section 4 introduces local likelihood density estimation. In Section 5 we use this density estimator to propose a novel entropy and mutual information estimator, and summarize certain theoretical properties of those estimator, which are then proved in Section 6. Section 7 provides numerical experiments demonstrating the superiority of the proposed estimator. We conclude the paper with a brief survey of related work followed by the discussion of our main results and some open problems.

## 2 Formal Problem Definition

In this section we briefly review the formal definition of Shannon entropy and mutual information, before formally defining the objective of our paper.

**Definition 1** Let  $\mathbf{x}$  denote a  $d$ -dimensional absolutely continuous random variable with probability density function  $f : \mathbb{R}^d \rightarrow \mathbb{R}$ . The Shannon differential entropy is defined as

$$H(\mathbf{x}) = - \int_{\mathbb{R}^d} f(\mathbf{x}) \log f(\mathbf{x}) d\mathbf{x} \quad (1)$$

**Definition 2** Let  $\mathbf{x}$  and  $\mathbf{y}$  denote  $d$ -dimensional and  $b$ -dimensional absolutely continuous random variables with probability density function  $f_X : \mathbb{R}^d \rightarrow \mathbb{R}$  and  $f_Y : \mathbb{R}^b \rightarrow \mathbb{R}$ , respectively. Let  $f_{XY}$  denote the joint probability density function of  $\mathbf{x}$  and  $\mathbf{y}$ . The mutual information between  $\mathbf{x}$  and  $\mathbf{y}$  is defined as

$$I(\mathbf{x} : \mathbf{y}) = \int_{\mathbf{y} \in \mathbb{R}^b} \int_{\mathbf{x} \in \mathbb{R}^d} f_{XY}(\mathbf{x}, \mathbf{y}) \log \frac{f_{XY}(\mathbf{x}, \mathbf{y})}{f_X(\mathbf{x}) f_Y(\mathbf{y})} d\mathbf{x} d\mathbf{y} \quad (2)$$

It is easy to show that

$$I(\mathbf{x} : \mathbf{y}) = H(\mathbf{x}) + H(\mathbf{y}) - H(\mathbf{x}, \mathbf{y}), \quad (3)$$

where  $H(\mathbf{x}, \mathbf{y})$  stands for the joint entropy of  $(\mathbf{x}, \mathbf{y})$ , and can be calculated from Eq. 1 using the joint density  $f_{XY}$ . We use the natural logarithms so that information is measured in nats.

It is sometime useful to represent entropy and mutual information as the following expectations:

$$H(\mathbf{x}) = \mathbb{E}_X[-\log f(\mathbf{x})] \quad (4)$$

$$I(\mathbf{x} : \mathbf{y}) = \mathbb{E}_{XY} \left[ \log \frac{f_{XY}(\mathbf{x}, \mathbf{y})}{f_X(\mathbf{x}) f_Y(\mathbf{y})} \right] \quad (5)$$

Assume now we are given  $N$  i.i.d. samples  $(\mathcal{X}, \mathcal{Y}) = \{(\mathbf{x}, \mathbf{y})^{(i)}\}_{i=1}^N$  from the unknown joint distribution  $f_{XY}$ . Our goal is then to construct a mutual information estimator  $\hat{I}(\mathbf{x} : \mathbf{y})$  based on those samples.

## 3 Limitations of Nonparametric MI Estimators

As pointed out in Section 1, one of the most popular class of mutual information estimators is based on  $k$ -nearest neighbor (kNN) graphs and their generalizations (Singh et al., 2003; Kraskov et al., 2004; Pál et al., 2010). However, it was recently shown that for strongly dependent variables, those estimators tend to underestimate the mutual information (Gao et al., 2015). To understand this problem, let us focus on kNN-based estimator as an example. The kNN estimator assumes uniform density within the kNN rectangle (containing  $k$ -nearest neighbors), as shown in Figure 1(a). Generally speaking, this assumption can be made valid for any relationship as long as we have sufficient number of samples. However, for limited sample size, this assumption becomes problematic when the relationship between the two variables becomes sufficiently strong. In fact, as shown in Fig. 1(b), the obtained *local* neighborhood induced by kNN is beyond the support of the probability distribution (shaded area).

This undesired behavior is closely related to the so-called *boundary effect* that occurs in nonparametric density estimation problem. Namely, for strongly dependent random variables, almost all the sample points are close to the boundary of the support (as illustrated in Figure 1(b)), making the density estimation problem difficult.

To relax the local uniformity assumption in kNN-based estimators, (Gao et al., 2015) proposed to replace the axis-aligned rectangle with a PCA-aligned rectangle *locally*, and use the volume of this rectangle for estimating the unknown density at a given point. Mathematically, the above revision was implemented by introducing a novel term that accounted for local non-uniformity. It was shown the the revised estimator significantly outperformed the existing estimators for strongly dependent variables. Nevertheless, the estimator suggested in (Gao et al., 2015) relied on a heuristic for determining when to use the correction term, and did not have any theoretical guarantees. In the remaining of this paper, we suggest a novel estimator based on *local gaussian approximation*, as more general approach to overcome the above limitations. The main idea is that, instead

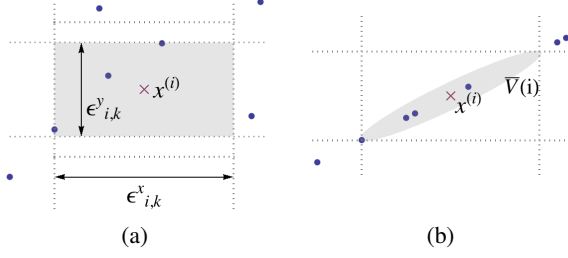


Figure 1: For a given sample point  $\mathbf{x}^{(i)}$ , we show the max-norm rectangle containing  $k$  nearest neighbors (a) for points drawn from a uniform distribution,  $k = 3$ , (shaded area), and (b) for points drawn from a distribution over two strongly correlated variables,  $k = 4$ , (the area within dotted lines).

of assuming a uniform distribution around the local kNN- or a PCA-aligned rectangle, we approximate the unknown density at each sample point by a local *Gaussian* distribution, which is estimated using the  $k$ -nearest neighborhood of that point. In addition to demonstrating superior empirical performance of the proposed estimator, we also show that it is asymptotically unbiased.

#### 4 Local Gaussian Density Estimation

In this section, we introduce a density estimation method called local Gaussian density estimation, or LGDE (Hjort and Jones, 1996), which serves as the basic building block for the proposed mutual information estimator.

Consider  $N$  i.i.d. samples  $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N$  drawn from an unknown density  $f(\mathbf{x})$ , where  $\mathbf{x}$  is a  $d$ -dimensional continuous random variable. The central idea behind LGDE is to locally approximate the unknown probability density at point  $\mathbf{x}$  using a Gaussian parametric family  $\mathcal{N}_d(\boldsymbol{\mu}(\mathbf{x}), \boldsymbol{\Sigma}(\mathbf{x}))$ , where  $\boldsymbol{\mu}(\mathbf{x})$  and  $\boldsymbol{\Sigma}(\mathbf{x})$  are the ( $\mathbf{x}$ -dependent) mean and covariance matrix of each local approximation. This intuition is formalized in the following definition:

**Definition 3 (Local Gaussian Density Estimator)** Let  $\mathbf{x}$  denote a  $d$ -dimensional absolutely continuous random variable with probability density function  $f(\mathbf{x})$ , and let  $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$  be  $N$  i.i.d. samples drawn from  $f(\mathbf{x})$ . Furthermore, let  $\mathbf{K}_{\mathbf{H}}(\mathbf{x})$  be a product kernel with diagonal bandwidth matrix  $\mathbf{H} = \text{diag}(h_1, h_2, \dots, h_d)$ , so that  $\mathbf{K}_{\mathbf{H}}(\mathbf{x}) = h_1^{-1}K(h_1^{-1}x_1)h_2^{-1}K(h_2^{-1}x_2)\dots h_d^{-1}K(h_d^{-1}x_d)$ , where  $K(\cdot)$  can be any one-dimensional kernel function. Then the Local Gaussian Density Estimator, or LGDE, of  $f(\mathbf{x})$  is given by

$$\hat{f}(\mathbf{x}) = \mathcal{N}_d(\mathbf{x}; \boldsymbol{\mu}(\mathbf{x}), \boldsymbol{\Sigma}(\mathbf{x})), \quad (6)$$

Here  $\boldsymbol{\mu}, \boldsymbol{\Sigma}$  are different for each point  $\mathbf{x}$ , and are obtained

by solving the following optimization problem,

$$\boldsymbol{\mu}(\mathbf{x}), \boldsymbol{\Sigma}(\mathbf{x}) = \arg \max_{\boldsymbol{\mu}, \boldsymbol{\Sigma}} \mathcal{L}(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma}), \quad (7)$$

where  $\mathcal{L}(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$  is the local likelihood function defined as follows:

$$\begin{aligned} \mathcal{L}(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) &= \frac{1}{N} \sum_{i=1}^N \mathbf{K}_{\mathbf{H}}(\mathbf{x}_i - \mathbf{x}) \log \mathcal{N}_d(\mathbf{x}_i; \boldsymbol{\mu}, \boldsymbol{\Sigma}) \\ &- \int \mathbf{K}_{\mathbf{H}}(\mathbf{t} - \mathbf{x}) \mathcal{N}_d(\mathbf{t}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) d\mathbf{t} \end{aligned} \quad (8)$$

The first term in the right hand side of Eq. 8 is the localized version of Gaussian log-likelihood. One can see that without the kernel function, Eq. 8 becomes similar to the global log-likelihood function of the Gaussian parametric family. However, since we do not have sufficient information to specify a global distribution, we make a local smoothness assumption by adding this kernel function. The second term of right hand side in Eq. 8 is a penalty term to ensure the consistency of the density estimator.

The key difference between kNN density estimator and LGDE is that the former assumes that the density is locally uniform over the neighborhood of each sample point, whereas the latter method relaxes *local uniformity* to *local linearity*<sup>1</sup>, which allows to compensate for the boundary bias. In fact, any non-uniform parametric probability distribution is suitable for fitting a local distribution under the local likelihood, and the Gaussian distribution used here is simply one realization.

I Theorem 1 below establishes the consistency property of this local Gaussian estimator; for a detailed proof see (Hjort and Jones, 1996).

**Theorem 1 (Hjort and Jones, 1996)** Let  $\mathbf{x}$  denote a  $d$ -dimensional absolutely continuous random variable with probability density function  $f(\mathbf{x})$ , and let  $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$  be  $N$  i.i.d. samples drawn from  $f(\mathbf{x})$ . Let  $\hat{f}(x)$  be the Local Gaussian Density Estimator with diagonal bandwidth matrix  $\text{diag}(h_1, h_2, \dots, h_d)$ , where the diagonal elements  $h_i$ -s satisfy the following conditions:

$$\lim_{N \rightarrow \infty} h_i = 0, \quad \lim_{N \rightarrow \infty} N h_i = \infty, \quad i = 1, 2, \dots, d. \quad (9)$$

Then the following holds:

$$\lim_{N \rightarrow \infty} \mathbb{E}|\hat{f}(\mathbf{x}) - f(\mathbf{x})| = 0 \quad (10)$$

$$\lim_{N \rightarrow \infty} \mathbb{E}|\hat{f}(\mathbf{x}) - f(\mathbf{x})|^2 = 0 \quad (11)$$

<sup>1</sup>To elaborate on the local linearity, we note that Gaussian distribution is essentially a special case of Elliptical distribution  $f(\mathbf{x}) = k * g((\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu}))$ . Therefore, the local Gaussian approximation actually assumes a rotated hyper-ellipsoid locally at each point.

The above theorem states that LGDE is asymptotically unbiased and L2-consistent.

## 5 LGDE-based Estimators for Entropy and Mutual Information

We now introduce our estimators for entropy and mutual information that are inspired by the local density estimation approach defined in the previous section.

Let us again consider  $N$  i.i.d samples  $(\mathcal{X}, \mathcal{Y}) = \{(\mathbf{x}, \mathbf{y})^{(i)}\}_{i=1}^N$  drawn from an unknown joint distribution  $f_{XY}$ , where  $\mathbf{x}$  and  $\mathbf{y}$  are random vectors of dimensionality  $d$  and  $b$ , respectively. Let us construct the following estimators for entropy,

$$\widehat{H}(\mathbf{x}) = -\frac{1}{N} \sum_{i=1}^N \log \widehat{f}(\mathbf{x}_i), \quad (12)$$

and mutual information

$$\widehat{I}(\mathbf{x} : \mathbf{y}) = \frac{1}{N} \sum_{i=1}^N \log \frac{\widehat{f}(\mathbf{x}_i, \mathbf{y}_i)}{\widehat{f}(\mathbf{x}_i) \widehat{f}(\mathbf{y}_i)} \quad (13)$$

where  $\widehat{f}(\mathbf{x})$ ,  $\widehat{f}(\mathbf{y})$ ,  $\widehat{f}(\mathbf{x}, \mathbf{y})$  are the local Gaussian density estimators for  $f_X(\mathbf{x})$ ,  $f_Y(\mathbf{y})$ ,  $f_{XY}(\mathbf{x}, \mathbf{y})$  respectively, defined in the previous section.

Recall that the entropy and mutual information can be written as appropriately defined expectations; see Eqs. 4 and 5. Then the proposed estimator simply replaces the expectation by the sample averages, and then plugs in density estimators from Section 4 into those expectations.

The next two theorems state that the proposed estimators are asymptotically unbiased.

**Theorem 2 (Asymptotic Unbiasedness of Entropy Estimator)**  
If the conditions in Eq. 9 hold, then the entropy estimator given by Eq. 12 is asymptotically unbiased, i.e.,

$$\lim_{N \rightarrow \infty} \mathbb{E} \widehat{H}(\mathbf{x}) = H(\mathbf{x}) \quad (14)$$

**Theorem 3 (Asymptotic Unbiasedness of MI Estimator)**  
If the conditions in Eq. 9 hold, then the mutual information estimator given by Eq. 13 is asymptotically unbiased:

$$\lim_{N \rightarrow \infty} \mathbb{E} \widehat{I}(\mathbf{x} : \mathbf{y}) = I(\mathbf{x} : \mathbf{y}) \quad (15)$$

We provide the proofs of the above theorems in the next section.

## 6 Proofs of the Theorems

Before getting to the actual proofs, we first introduce the Lebesgue's dominated convergence theorem.

**Theorem 4 (Lebesgue dominated convergence theorem)**  
Let  $\{f_N\}$  be a sequence of functions, and assume this sequence converges point-wise to a function  $f$ , i.e.,  $f_N(\mathbf{x}) \rightarrow f(\mathbf{x})$  for any  $\mathbf{x} \in \mathbb{R}^d$ . Furthermore, let us assume that  $f_N$  is dominated by an integrable function  $g$ , e.g., we have for any  $\mathbf{x}$

$$|f_N(\mathbf{x})| \leq g(\mathbf{x})$$

Then we have

$$\lim_{N \rightarrow \infty} \int_{\mathbf{x} \in \mathbf{X}} |f_N(\mathbf{x}) - f(\mathbf{x})| d\mathbf{x} = 0$$

### 6.1 Proof of Theorem 2

Consider  $N$  i.i.d. samples  $\{\mathbf{x}^{(i)}\}_{i=1}^N$  drawn from the probability density  $f(\mathbf{x})$ , and let  $F_N(\mathbf{x})$  denote the empirical cumulative distribution function.

Let us define the following two quantities:

$$H_1 = -\frac{1}{N} \sum_{i=1}^N \ln \mathbb{E} \widehat{f}(\mathbf{x}_i) \quad (16)$$

$$H_2 = -\frac{1}{N} \sum_{i=1}^N \ln f(\mathbf{x}_i) \quad (17)$$

Then we have,

$$\begin{aligned} & \mathbb{E} |\widehat{H}(\mathbf{x}) - H(\mathbf{x})| \\ &= \mathbb{E} |(\widehat{H} - H_1) + (H_1 - H_2) + (H_2 - H)| \\ &\leq \mathbb{E} |\widehat{H} - H_1| + \mathbb{E} |H_1 - H_2| + \mathbb{E} |H_2 - H| \end{aligned} \quad (18)$$

We now proceed to show that each of the terms in Eq. 18 individually converges to 0 in the limit  $N \rightarrow \infty$ , which will then yield Eq. 14.

First, we note that according to the mean value theorem, for any  $\mathbf{x}$ , there exist  $t_{\mathbf{x}}$  and  $t'_{\mathbf{x}}$  in  $(0, 1)$ , such that

$$\begin{aligned} \ln \widehat{f}(\mathbf{x}) &= \ln \mathbb{E} \widehat{f}(\mathbf{x}) + \\ & (\widehat{f}(\mathbf{x}) - \mathbb{E} \widehat{f}(\mathbf{x})) \ln (t_{\mathbf{x}} \widehat{f}(\mathbf{x}) + (1 - t_{\mathbf{x}}) \mathbb{E} \widehat{f}(\mathbf{x})) \end{aligned} \quad (19)$$

and

$$\begin{aligned} \ln \mathbb{E} \widehat{f}(\mathbf{x}) &= \ln f(\mathbf{x}) + \\ & (\mathbb{E} \widehat{f}(\mathbf{x}) - f(\mathbf{x})) \ln (t'_{\mathbf{x}} f(\mathbf{x}) + (1 - t'_{\mathbf{x}}) \mathbb{E} \widehat{f}(\mathbf{x})) \end{aligned} \quad (20)$$

For the first term in Eq. 18, we use Eq. 19 to obtain

$$\begin{aligned}
& \mathbb{E} \left| \widehat{H} - H_1 \right| \\
&= \mathbb{E} \left| \int [\ln \widehat{f}(\mathbf{x}) - \ln \mathbb{E} \widehat{f}(\mathbf{x})] dF_N(\mathbf{x}) \right| \\
&= \mathbb{E} \left| \int \frac{|\widehat{f}(\mathbf{x}) - \mathbb{E} \widehat{f}(\mathbf{x})|}{t_{\mathbf{x}} \widehat{f}(\mathbf{x}) + (1 - t_{\mathbf{x}}) \mathbb{E} \widehat{f}(\mathbf{x})} dF_N(\mathbf{x}) \right| \\
&\leq \frac{1}{1-t} \mathbb{E} \left| \int \frac{|\widehat{f}(\mathbf{x}) - \mathbb{E} \widehat{f}(\mathbf{x})|}{\mathbb{E} \widehat{f}(\mathbf{x})} dF_N(\mathbf{x}) \right| \\
&= \frac{1}{1-t} \mathbb{E} \left( \frac{1}{N} \sum_{i=1}^N \frac{|\widehat{f}(\mathbf{x}_i) - \mathbb{E} \widehat{f}(\mathbf{x}_i)|}{\mathbb{E} \widehat{f}(\mathbf{x}_i)} \right) \\
&= \frac{1}{1-t} \mathbb{E} \left( \mathbb{E} \left( \frac{|\widehat{f}(\mathbf{u}) - \mathbb{E} \widehat{f}(\mathbf{u})|}{\mathbb{E} \widehat{f}(\mathbf{u})} \right) \middle| \mathbf{x} = \mathbf{u} \right) \\
&= \frac{1}{1-t} \int |\widehat{f}(\mathbf{u}) - \mathbb{E} \widehat{f}(\mathbf{u})| \frac{\widehat{f}(\mathbf{u})}{\mathbb{E} \widehat{f}(\mathbf{u})} d\mathbf{u} \quad (21)
\end{aligned}$$

where  $t$  is the maximum value among all  $t_{\mathbf{x}}$ . Using Theorem 1, we have  $|\widehat{f}(\mathbf{u}) - \mathbb{E} \widehat{f}(\mathbf{u})| \rightarrow 0$  as  $N \rightarrow \infty$ . Furthermore, it is possible to show that  $\exists N_0$ , so that for any  $N > N_0$  one has  $|\widehat{f}(\mathbf{u}) - \mathbb{E} \widehat{f}(\mathbf{u})| \frac{\widehat{f}(\mathbf{u})}{\mathbb{E} \widehat{f}(\mathbf{u})} < 2f(\mathbf{u})$ . Thus, using Theorem 4, we obtain

$$\lim_{N \rightarrow \infty} \mathbb{E} |H - H_1| = 0 \quad (22)$$

Similarly, using Eq. 20,  $\mathbb{E} |H_1 - H_2|$  can be written as

$$\begin{aligned}
& \mathbb{E} |H_1 - H_2| \\
&= \mathbb{E} \left| \int [\ln \mathbb{E} \widehat{f}(\mathbf{x}) - \ln f(\mathbf{x})] dF_N(\mathbf{x}) \right| \\
&= \mathbb{E} \left| \int \frac{|\mathbb{E} \widehat{f}(\mathbf{x}) - f(\mathbf{x})|}{t'_{\mathbf{x}} f(\mathbf{x}) + (1 - t'_{\mathbf{x}}) \mathbb{E} \widehat{f}(\mathbf{x})} dF_N(\mathbf{x}) \right| \\
&\leq \frac{1}{t'} \mathbb{E} \left| \int \frac{|\mathbb{E} \widehat{f}(\mathbf{x}) - f(\mathbf{x})|}{f(\mathbf{x})} dF_N(\mathbf{x}) \right| \\
&= \frac{1}{t'} \mathbb{E} \left( \frac{1}{N} \sum_{i=1}^N \frac{|\mathbb{E} \widehat{f}(\mathbf{x}_i) - f(\mathbf{x}_i)|}{f(\mathbf{x}_i)} \right) \\
&= \frac{1}{t'} \int f(\mathbf{x}) \frac{|\mathbb{E} \widehat{f}(\mathbf{x}) - f(\mathbf{x})|}{f(\mathbf{x})} d\mathbf{x} \\
&= \frac{1}{t'} \int |\mathbb{E} \widehat{f}(\mathbf{x}) - f(\mathbf{x})| d\mathbf{x} \quad (23)
\end{aligned}$$

where  $t'$  is the minimum value among all  $t'_{\mathbf{x}}$ .

Invoking Theorem 1 again, we observe that the last term in Eq. 23  $|\mathbb{E} \widehat{f}(\mathbf{x}) - f(\mathbf{x})| \rightarrow 0$  as  $N \rightarrow \infty$ , and is bounded by  $2f(\mathbf{x})$  for sufficiently large  $N$  (e.g., when  $\widehat{f}(\mathbf{u})$  and  $\mathbb{E} \widehat{f}(\mathbf{u})$  are sufficiently close). Therefore, by Theorem 4, we have

$$\lim_{N \rightarrow \infty} \mathbb{E} |H_1 - H_2| = 0 \quad (24)$$

Finally, for the last term in Eq. 18, we note that

$$\mathbb{E} H_2 = -\frac{1}{N} \mathbb{E} \sum_{i=1}^N \ln f(\mathbf{x}_i) = \mathbb{E} [-\ln f(\mathbf{x})] \quad (25)$$

Thus,  $\mathbb{E} H_2$  is simply the entropy in Definition 1; see Eq. 4. Therefore,

$$\lim_{N \rightarrow \infty} \mathbb{E} |H_2 - H| = 0 \quad (26)$$

Combining Eqs. 22, 24, 26 and 18, we arrive at Eq. 14, which concludes the proof.

## 6.2 Proof of Theorem 3

For mutual information estimation, we use Eq. 3 to get

$$\begin{aligned}
\mathbb{E} |\widehat{I}(\mathbf{x} : \mathbf{y}) - I(\mathbf{x} : \mathbf{y})| &\leq \mathbb{E} |H(\mathbf{x}) - \widehat{H}(\mathbf{x})| \\
&\quad + \mathbb{E} |H(\mathbf{y}) - \widehat{H}(\mathbf{y})| \\
&\quad + \mathbb{E} |H(\mathbf{x}, \mathbf{y}) - \widehat{H}(\mathbf{x}, \mathbf{y})| \quad (27)
\end{aligned}$$

Using Theorem 2, we see that all three terms on the right hand side in Eq. 27 converge to zero as  $N \rightarrow \infty$ , therefore  $\lim_{N \rightarrow \infty} \mathbb{E} |\widehat{I}(\mathbf{x} : \mathbf{y}) - I(\mathbf{x} : \mathbf{y})| = 0$ , thus concluding the proof.

## 7 Experiments

### 7.1 Implementation Details

Our main computational task is to maximize the local likelihood function in Eq. 8. Since computing the second term on the right hand side of Eq. 8 requires integration that can be time-consuming, we choose the kernel function  $K(\cdot)$  to be a Gaussian kernel,  $\mathbf{K}_{\mathbf{H}}(\mathbf{t} - \mathbf{x}) = \mathcal{N}_d(\mathbf{t}; \mathbf{x}, \mathbf{H})$  so that the integral can be performed analytically, yielding

$$\int \mathbf{K}_{\mathbf{H}}(\mathbf{t} - \mathbf{x}) \mathcal{N}_d(\mathbf{t}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) d\mathbf{t} = \mathcal{N}_d(\mathbf{x}; \boldsymbol{\mu}, \mathbf{H} + \boldsymbol{\Sigma}) \quad (28)$$

Thus, Eq. 8 reduces to

$$\begin{aligned}
\mathcal{L}(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) &= \frac{1}{N} \sum_{i=1}^N \mathcal{N}_d(\mathbf{x}_i; \mathbf{x}, \mathbf{H}) \log \mathcal{N}_d(\mathbf{x}_i; \boldsymbol{\mu}, \boldsymbol{\Sigma}) \\
&\quad - \mathcal{N}_d(\mathbf{x}; \boldsymbol{\mu}, \mathbf{H} + \boldsymbol{\Sigma}) \quad (29)
\end{aligned}$$

Maximizing Eq. 29 is a constrained non-convex optimization problem with the condition that the covariance matrix  $\boldsymbol{\Sigma}$  is positive semi-definite. We use Cholesky parameterization to enforce the positive semi-definiteness of  $\boldsymbol{\Sigma}$ , which allows to reduce our constrained optimization problem into an unconstrained one. Also, since we would like to preserve the local structure of the data, we select the bandwidth to be close to the distance between pair of k-nearest points (averaged over all the points).

We use Newton-Raphson method to do the maximization although the function itself is not exactly concave. The full algorithm for our estimator is given in Algorithm 1 which takes Algorithm 2 as a subroutine. Note that in Algorithm 2, the Wolfe condition is a set of inequalities in performing quasi-Newton methods (Wolfe, 1969).

---

### Algorithm 1 Mutual Information Estimation with Local Gaussian Approximation

---

**Input:** points  $(\mathbf{x}, \mathbf{y})^{(1)}, (\mathbf{x}, \mathbf{y})^{(2)}, \dots, (\mathbf{x}, \mathbf{y})^{(N)}$   
**Output:**  $\hat{I}(\mathbf{x}; \mathbf{y})$   
 Calculate entropy  $\hat{H}(\mathbf{x})$  using samples  $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(N)}$   
 Calculate entropy  $\hat{H}(\mathbf{y})$  using samples  $\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \dots, \mathbf{y}^{(N)}$   
 Calculate joint entropy  $\hat{H}(\mathbf{x}, \mathbf{y})$  using input samples  $(\mathbf{x}, \mathbf{y})^{(1)}, (\mathbf{x}, \mathbf{y})^{(2)}, \dots, (\mathbf{x}, \mathbf{y})^{(N)}$   
 Return estimated mutual information  $\hat{I} = \hat{H}(\mathbf{x}) + \hat{H}(\mathbf{y}) - \hat{H}(\mathbf{x}, \mathbf{y})$

---



---

### Algorithm 2 Entropy Estimation with Local Gaussian Approximation

---

**Input:** points  $\mathbf{u}^{(1)}, \mathbf{u}^{(2)}, \dots, \mathbf{u}^{(N)}$   
**Output:**  $\hat{H}(\mathbf{u})$   
 Initialize  $\hat{H}(\mathbf{u}) = 0$   
**for** each point  $\mathbf{x}^{(i)}$  **do**  
   initialize  $\boldsymbol{\mu} = \boldsymbol{\mu}_0, \mathbf{L} = \mathbf{L}_0$   
   **while** not  $\mathcal{L}(\mathbf{x}^{(i)}, \boldsymbol{\mu}, \boldsymbol{\Sigma} = \mathbf{L} * \mathbf{L}^T)$  converge **do**  
     Calculate  $\mathcal{L}(\mathbf{x}^{(i)}, \boldsymbol{\mu}, \boldsymbol{\Sigma} = \mathbf{L} * \mathbf{L}^T)$   
     Calculate gradient vector  $\mathbf{G}$  of  $\mathcal{L}(\mathbf{x}^{(i)}, \boldsymbol{\mu}, \boldsymbol{\Sigma} = \mathbf{L} * \mathbf{L}^T)$ , with respect to  $\boldsymbol{\mu}, \mathbf{L}$   
     Calculate Hessian matrix of  $\mathbf{H}$  of  $\mathcal{L}(\mathbf{x}^{(i)}, \boldsymbol{\mu}, \boldsymbol{\Sigma} = \mathbf{L} * \mathbf{L}^T)$ , with respect to  $\boldsymbol{\mu}, \mathbf{L}$   
     Do Hessian modification to ensure the positive semi-definiteness of  $\mathbf{H}$   
     Calculate descent direction  $\mathbf{D} = -\alpha \mathbf{H}^{-1} \mathbf{G}$ , where we compute  $\alpha$  to satisfy Wolfe condition  
     Update  $\boldsymbol{\mu}, \mathbf{L}$  with  $(\boldsymbol{\mu}, \mathbf{L}) + \mathbf{D}$   
   **end while**  
    $\hat{f}(\mathbf{x}^{(i)}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma} = \mathbf{L} * \mathbf{L}^T)$   
    $\hat{H}(\mathbf{u}) = \hat{H}(\mathbf{u}) - \frac{\log \hat{f}(\mathbf{x}^{(i)})}{N}$   
**end for**

---

In a single step, evaluating the gradient and Hessian in Algorithm 2 would take  $O(N)$  time because Eq. 8 is a summation over all the points. However, for points that are far from the current point  $\mathbf{x}^{(i)}$ , the kernel weight function is very close to zero and we can ignore those point and do the summation only over a local neighborhood of  $\mathbf{x}^{(i)}$ .

## 7.2 Experiments with synthetic data

**Functional relationships** We test our MI estimator for near-functional relationships of form  $Y = f(X) + \mathcal{U}(0, \theta)$ ,

where  $\mathcal{U}(0, \theta)$  is the uniform distribution over the interval  $(0, \theta)$ , and  $X$  is drawn randomly uniformly from  $[0, 1]$ . Similar relationships were studied in (Reshef et al., 2011), (Kinney and Atwal, 2014) and (Gao et al., 2015).

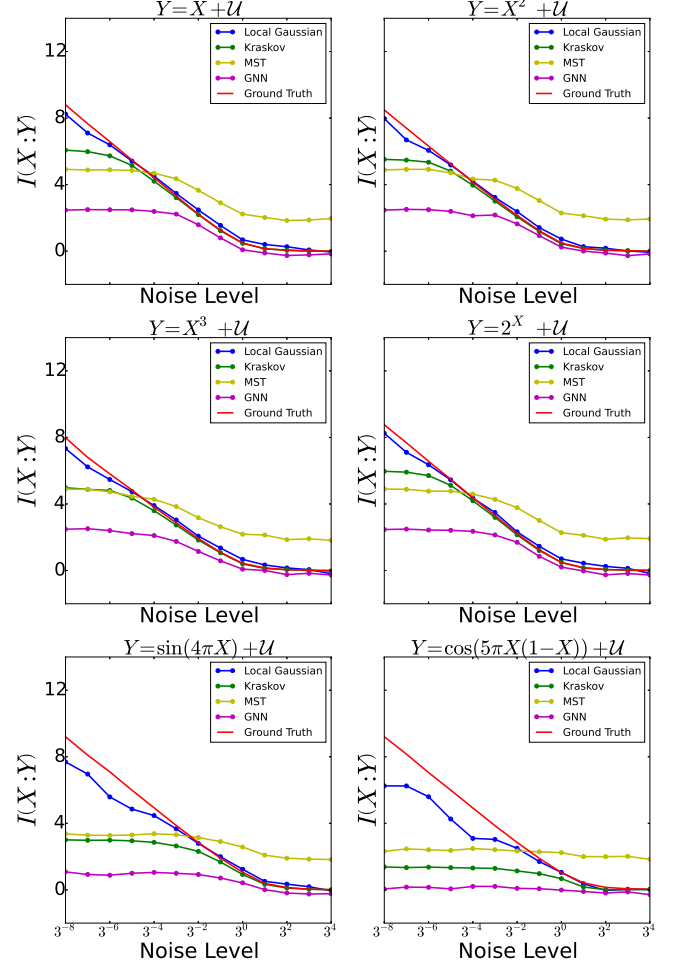


Figure 2: Functional relationship test for mutual information estimators. The horizontal axis is the value of  $\theta$  which controls the noise level; the vertical axis is the mutual information in nats. For the Kraskov and GNN estimators we used nearest neighbor parameter  $k = 5$ . For the local Gaussian estimator, we choose the bandwidth to be the distance between a point and its 5rd nearest neighbor.

We compare our estimator to several baselines that include the kNN estimator proposed by (Kraskov et al., 2004), an estimator based on generalized nearest-neighbor graphs (GNN) (Pál et al., 2010), and minimum spanning tree method (MST) (Yukich and Yukich, 1998). We evaluate those estimators for six different functional relationships as indicated in Figure 2. We use  $N = 2500$  sample points for each relationship. To speed up the optimization, we limited the summation in Eq. 29 to only  $k$  nearest neighbors, thus reducing the computational complexity from  $O(N)$  to  $O(k)$  in every iteration step of Algorithm 2.

One can see from Fig. 2 that when  $\theta$  is relatively large, all methods except MST produce accurate estimates of MI. However, as one decreases  $\theta$ , all three baseline estimators start to significantly underestimate mutual information. In this low-noise regime, our proposed estimator outperforms the baselines, at times by a significant margin. Note also that all the estimators, including ours, perform relatively poorly for highly non-linear relationships (the last row in Figure 2). According to our intuition, this happens when the scale of the non-linearity becomes sufficiently small, so that the linear approximation of the relationship around the local neighborhood of each sample point does not hold. Under this scenario, accuracy can be recovered by adding more samples.

## 8 Related Work

**Mutual Information Estimators** Recently, there has been a significant amount of work on estimating information-theoretic quantities such as entropy, mutual information, and divergences, from i.i.d. samples. Methods include k-nearest-neighbors (Singh et al., 2003), (Kraskov et al., 2004), (Pál et al., 2010), (Póczos et al., 2011); minimum spanning trees (Yukich and Yukich, 1998); kernel density estimate (Moon et al., 1995), (Singh and Poczos, 2014); maximum likelihood density ratio (Suzuki et al., 2008); ensemble methods (Moon and Hero, 2014), (Sricharan et al. (2013), etc. As pointed out earlier, all of those methods underestimate the mutual information when two variables have strong dependency. (Gao et al., 2015) addressed this shortcoming by introducing a local non-uniformity correction, but their estimator depended on a heuristically defined threshold parameter and lacked performance guarantees.

**Density Estimation and Boundary Bias** Density estimation is a classic problem in statistics and machine learning. Kernel density estimation and k-nearest-neighbor density estimates are the two most popular and successful non-parametric methods. However, it has been recognized that these non-parametric techniques often suffer from the problem of so-called “boundary bias”. Researchers have proposed a variety of methods to overcome the bias, such as the reflection method (Schuster, 1985), (Silverman, 1986); the boundary kernel method (Zhang and Karunamuni, 2000), the transformation method (Marron and Ruppert, 1994), the pseudo-data method (Cowling and Hall, 1996) and others. All these methods are useful in some particular settings. But when it comes to mutual information estimation, how can we choose the most efficient one to use? It seems that local likelihood method (Hjort and Jones, 1996), (Loader, 1996), is a good choice for estimating the mutual information due to its ability to detect the boundary without any prior knowledge. Previous studies have already proven the power of *local regression*, which can automatically overcome the boundary bias. Methods based on *local likelihood* estimation has traditionally at-

tracted less attention due to their computational complexity. However, advances in computational power allow us to re-consider this class of method.

## 9 Conclusion and Future Work

Past research on mutual information estimation has mostly focused on distinguishing weak dependence from independence. However, in the era of big data, we are often interested in highlighting the strongest dependencies among a large number of variables. When those variables are highly inter-dependent, traditional non-parametric mutual information estimators fail to accurately estimate the value due to the boundary bias.

We have addressed this shortcoming by introducing a novel semi-parametric method for estimating entropy and mutual information based on local Gaussian approximation of the unknown density at the sample points. We demonstrated that the proposed estimators are asymptotically unbiased. We also showed empirically that the proposed estimator has a superior performance compared to a number of popular baseline methods, and can accurately measure strength of the relationship even for strongly dependent variables, and limited number of samples.

There are several potential avenues for future work. First of all, we would like to validate the proposed estimator in higher-dimensional settings. In principle, the approach is general and can be applied in any dimensions. However, the optimization procedure may be computationally expensive in higher dimensions, since the number of parameters scales as  $O(d^2)$  with dimensionality  $d$ . An intuitive solution would be to initialize the parameters with the results obtained from the close points, which can facilitate convergence.

Another interesting issue is the bandwidth selection, which is an important problem in general density estimation problems. If the bandwidth is too large, the local Gaussian assumption may not be valid, whereas very small bandwidth will result in non-smooth densities. Ideally, we would like to choose the bandwidth in a way that preserves the local Gaussian structure in the neighborhood of each point. Another interesting extension would be choosing the bandwidth adaptively for each point.

Finally, while here we have focused on the asymptotic unbiasedness of the proposed estimator, it will be very valuable to establish theoretical results about the convergence rates of the estimators, as well as its variance in the large sample limit.

## Acknowledgements

This research was supported in part by DARPA grant No. W911NF-12-1-0034.

## References

- Ann Cowling and Peter Hall. On pseudodata methods for removing boundary effects in kernel density estimation. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 551–563, 1996.
- Shuyang Gao, Greg Ver Steeg, and Aram Galstyan. Efficient estimation of mutual information for strongly dependent variables. In *AISTATS'15*, 2015.
- NL Hjort and MC Jones. Locally parametric nonparametric density estimation. *The Annals of Statistics*, pages 1619–1647, 1996.
- J. Kinney and G. Atwal. Equitability, mutual information, and the maximal information coefficient. *Proceedings of the National Academy of Sciences*, 111(9):3354–3359, 2014.
- A. Kraskov, H. Stögbauer, and P. Grassberger. Estimating mutual information. *Phys. Rev. E*, 69:066138, 2004. doi: 10.1103/PhysRevE.69.066138. URL <http://link.aps.org/doi/10.1103/PhysRevE.69.066138>.
- Clive R Loader. Local likelihood density estimation. *The Annals of Statistics*, 24(4):1602–1618, 1996.
- James Stephen Marron and David Ruppert. Transformations to reduce boundary bias in kernel density estimation. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 653–671, 1994.
- K.R. Moon and A.O. Hero. Ensemble estimation of multivariate f-divergence. In *Information Theory (ISIT), 2014 IEEE International Symposium on*, pages 356–360, June 2014. doi: 10.1109/ISIT.2014.6874854.
- Young-Il Moon, Balaji Rajagopalan, and Upmanu Lall. Estimation of mutual information using kernel density estimators. *Physical Review E*, 52(3):2318–2321, 1995.
- Dávid Pál, Barnabás Póczos, and Csaba Szepesvári. Estimation of rényi entropy and mutual information based on generalized nearest-neighbor graphs. In *Advances in Neural Information Processing Systems 23*, pages 1849–1857. Curran Associates, Inc., 2010.
- Barnabás Póczos, Liang Xiong, and Jeff Schneider. Nonparametric divergence estimation with applications to machine learning on distributions. In *Proceedings of Uncertainty in Artificial Intelligence (UAI)*, 2011.
- David N Reshef, Yakir A Reshef, Hilary K Finucane, Sharon R Grossman, Gilean McVean, Peter J Turnbaugh, Eric S Lander, Michael Mitzenmacher, and Pardis C Sabeti. Detecting novel associations in large data sets. *science*, 334(6062):1518–1524, 2011.
- Eugene F Schuster. Incorporating support constraints into nonparametric estimators of densities. *Communications in Statistics-Theory and methods*, 14(5):1123–1136, 1985.
- C.E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379423, 1948.
- Bernard W Silverman. *Density estimation for statistics and data analysis*, volume 26. CRC press, 1986.
- Harshinder Singh, Neeraj Misra, Vladimir Hnizdo, Adam Fedorowicz, and Eugene Demchuk. Nearest neighbor estimates of entropy. *American Journal of Mathematical and Management Sciences*, 23(3-4):301–321, 2003. doi: 10.1080/01966324.2003.10737616. URL <http://dx.doi.org/10.1080/01966324.2003.10737616>.
- Shashank Singh and Barnabas Póczos. Generalized exponential concentration inequality for renyi divergence estimation. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 333–341, 2014. URL [http://machinelearning.wustl.edu/mlpapers/papers/icml2014c1\\_singh14](http://machinelearning.wustl.edu/mlpapers/papers/icml2014c1_singh14).
- K. Sricharan, D. Wei, and A.O. Hero. Ensemble estimators for multivariate entropy estimation. *Information Theory, IEEE Transactions on*, 59(7):4374–4388, July 2013. ISSN 0018-9448. doi: 10.1109/TIT.2013.2251456.
- Taiji Suzuki, Masashi Sugiyama, Jun Sese, and Takafumi Kanamori. Approximating mutual information by maximum likelihood density ratio estimation. In Yvan Saeys, Huan Liu, Iaki Inza, Louis Wehenkel, and Yves Van de Peer, editors, *FSDM*, volume 4 of *JMLR Proceedings*, pages 5–20. JMLR.org, 2008.
- Philip Wolfe. Convergence conditions for ascent methods. *SIAM review*, 11(2):226–235, 1969.
- Joseph E Yukich and Joseph Yukich. *Probability theory of classical Euclidean optimization problems*. Springer Berlin, 1998.
- Shunpu Zhang and Rohana J Karunamuni. On nonparametric density estimation at the boundary\*. *Journal of nonparametric statistics*, 12(2):197–221, 2000.

---

# Psychophysical Detection Testing with Bayesian Active Learning

---

**Jacob R. Gardner**

gardner.jake@wustl.edu  
Washington University in St. Louis  
St. Louis, MO 63130

**Xinyu Song**

xinyu.song@wustl.edu  
Washington University in St. Louis  
St. Louis, MO 63130

**Kilian Q. Weinberger**

kilian@wustl.edu  
Washington University in St. Louis  
St. Louis, MO 63130

**Dennis Barbour**

dbarbour@wustl.edu  
Washington University in St. Louis  
St. Louis, MO 63130

**John P. Cunningham**

jpc2181@columbia.edu  
Columbia University  
New York, NY 10027

## Abstract

Psychophysical detection tests are ubiquitous in the study of human sensation and the diagnosis and treatment of virtually all sensory impairments. In many of these settings, the goal is to recover, from a series of binary observations from a human subject, the latent function that describes the discriminability of a sensory stimulus over some relevant domain. The auditory detection test, for example, seeks to understand a subject’s likelihood of hearing sounds as a function of frequency and amplitude. Conventional methods for performing these tests involve testing stimuli on a pre-determined grid. This approach not only samples at very uninformative locations, but also fails to learn critical features of a subject’s latent discriminability function. Here we advance active learning with Gaussian processes to the setting of psychophysical testing. We develop a model that incorporates strong prior knowledge about the class of stimuli, we derive a sensible method for choosing sample points, and we demonstrate how to evaluate this model efficiently. Finally, we develop a novel likelihood that enables testing of multiple stimuli simultaneously. We evaluate our method in both simulated and real auditory detection tests, demonstrating the merit of our approach.

## 1 INTRODUCTION

Psychophysical tests are a fundamental tool for investigating human perception: does a particular stimulus produce sensation for a particular person? The most common form of psychophysical tests – *detection tests* – present  $n$  sensory stimuli to a subject, and ask for  $n$  binary reports as

to whether each stimulus was detected or not. Detection tests exist for vision (Schiefer et al., 2005), pain (Carter and Shieh, 2009), and many other settings. Perhaps the most common example is audiometry (Carhart and Jerger, 1959; Don et al., 1978; Hughson and Westlake, 1944): a subject is presented with a sequence of  $n$  tones  $\mathbf{x}_t \forall t = 1, \dots, n$ , where each tone  $\mathbf{x}_t \in \mathbb{R}^2$  is a pure tone with a specific frequency (pitch) and intensity (volume). The subject reports an observation  $y_t = 1$  if he/she heard the tone, and a  $y_t = 0$  is concluded in the absence of a positive report. The purpose of the test is to infer, from this sequence of observations, the underlying *audiometric function*  $g(\mathbf{x})$ , a function that describes how likely the subject is to hear sounds over the domain of typical frequencies and intensities. There is substantial variability in each person’s audiogram, particularly for those with partial, selective, or degenerative hearing loss (Gosztonyi Jr et al., 1971; Robinson, 1991; Schmuziger et al., 2004). Accurate estimates of audiograms are thus essential to understanding human audition, and to all medical studies and treatments of various forms of hearing loss.

A standard auditory detection test is carried out by playing an  $n$ -length sequence of pure tones on a pre-defined grid in frequency-intensity space. This approach, while simple, has several salient drawbacks that lead to an unnecessarily large  $n$ . First, a given tone is played multiple times, even if it is highly audible or highly inaudible. Second, information is not shared between previous outcomes. For example, human audition is monotonically increasing in intensity, but in the standard test, even if a particular frequency of sound is heard at a given intensity, tones with the same frequency but higher intensity will still be tested. Finally, owing to limitations on the size of sequence  $n$ , a standard detection test probes only six discrete frequencies (Madison et al., 2005). The coarseness of this grid can cause significant errors, as human hearing loss can span a range narrow enough to be entirely missed by these six frequen-



cies (Jerger, 1960; Zhao et al., 2002; Zhao and Stephens, 1998). All of these issues, combined with the impracticality and burden to human subjects of a large  $n$  sequence, motivate an active learning approach.

Here we treat psychophysical detection tests as an active learning problem, extending and adapting recent work on active learning with Gaussian processes (GPs) (Garnett et al., 2013; Houlsby et al., 2011; Iwata et al., 2013). Our method addresses all the drawbacks of grid-sampling by performing Bayesian active learning of the audiometric function  $g(\mathbf{x})$ . Specifically, we place a GP prior on the latent audiogram  $f(\mathbf{x})$ , which we transform to a  $[0, 1]$  valued quantity using a probit transformation (Kuss and Rasmussen, 2005), such that  $g(\mathbf{x}) \approx \Phi(f(\mathbf{x}))$ . We use this model to sequentially sample at each time step  $t$  the most informative next tones conditioned on the previous  $t-1$  observations  $y_1, \dots, y_{t-1}$ . This model significantly enhances the accuracy and efficiency of learning audiograms. Our work offers two main contributions:

1. We extend and adapt existing work on Bayesian optimization and active learning to the setting of psychophysical detection tests. We present a model that incorporates strong prior knowledge about the auditory stimulus space, and we present experimental results demonstrating the effectiveness of a Bayesian active learning approach.
2. We develop a novel ‘OR-channel’ likelihood that allows the query of *multiple tones simultaneously*. We analyze this likelihood in the active learning context, clarifying the non-obvious intuition for why and when such an approach can outperform single-tone queries.

We evaluate our algorithm on both simulated and real audiometric detection tests. Our active learning approach obtains finer grained estimates of the audiogram  $g(\mathbf{x})$  with substantially fewer stimuli queries (lower  $n$ ). We note that, in the remainder of this work (notably our experiments), we will continue to use the example and nomenclature of audiometry, though our algorithm is precisely equivalent for other psychophysical detection tests as well.

## 2 GAUSSIAN PROCESSES

Throughout this paper we will make extensive use of Gaussian processes (GPs). A GP is formally a prior over functions,  $f \sim \mathcal{GP}(\mu_0(\cdot), k(\cdot, \cdot))$ , parameterized by a mean function  $\mu_0(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})]$  and covariance function  $k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - \mu_0(\mathbf{x}))(f(\mathbf{x}') - \mu_0(\mathbf{x}'))]$ .

For any set of  $n$  observations  $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n]$ , the GP implies that their function values  $\mathbf{f} = [f(\mathbf{x}_1), \dots, f(\mathbf{x}_n)]^\top$  are jointly Gaussian distributed,  $\mathbf{f} \sim \mathcal{N}(\mu(\mathbf{X}), \mathbf{K})$ , where  $\mathbf{K}$  defines the covariance  $\mathbf{K}_{ij} = \text{Cov}[f_i, f_j] = k(\mathbf{x}_i, \mathbf{x}_j)$ .

If we add a test point  $\mathbf{x}^*$  with unknown function value  $f^*$  this distribution extends naturally by one dimension to

$$\begin{bmatrix} \mathbf{f} \\ f^* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu(\mathbf{X}) \\ \mu(\mathbf{x}^*) \end{bmatrix}, \begin{bmatrix} \mathbf{K} & \mathbf{k}^* \\ \mathbf{k}^{*\top} & k(\mathbf{x}^*, \mathbf{x}^*) \end{bmatrix}\right).$$

We can utilize standard Gaussian conditioning rules (Rasmussen and Williams, 2006) to derive the posterior distribution,  $p(f^*|\mathbf{X}, \mathbf{f}, \mathbf{x}^*)$ , which is Gaussian with mean and variance

$$\begin{aligned} \mu^*(\mathbf{x}^*) &= \mu_0(\mathbf{x}^*) + \mathbf{k}^{*\top} \mathbf{K}^{-1}(\mathbf{f} - \mu_0(\mathbf{x}^*)) \quad (1) \\ \sigma^{*2}(\mathbf{x}^*) &= k(\mathbf{x}^*, \mathbf{x}^*) - \mathbf{k}^{*\top} \mathbf{K}^{-1} \mathbf{k}^*. \quad (2) \end{aligned}$$

Here  $\mathbf{k}^* = [k(\mathbf{x}^*, \mathbf{x}_1), \dots, k(\mathbf{x}^*, \mathbf{x}_n)]^\top$  denotes the kernel vector between the test input  $\mathbf{x}^*$  and each training input.

In practice, we often do not observe  $f_i$  directly, but rather some dependent random variable  $y_i$ . A popular example is to assume additive Gaussian noise,  $y_i = f_i + \epsilon$  with  $\epsilon \sim \mathcal{N}(0, \sigma_n^2)$ . In this setting, the distribution for  $f^*$  remains Gaussian, with a mean and variance similar to eqs. (1) and (2) (where  $\mathbf{K}$  is replaced with  $\mathbf{K} + \sigma_n^2 \mathbf{I}$ ).

However, with most observation models, the posterior distribution of  $f^*$  conditioned on  $\mathbf{y}$  is not Gaussian, and exact inference becomes impossible. Approximate inference may be performed using a Gaussian approximation to the likelihood (Kuss and Rasmussen, 2005; Minka, 2001). In particular, by using a Gaussian approximation to the likelihood, we recover the Gaussianity of the posterior. For a full treatment of Gaussian processes, see (Rasmussen and Williams, 2006).

Note that in many cases, our goal is to make predictions, for which we use the posterior predictive distribution—a distribution over  $y^*$ :

$$p(y^*|\mathbf{X}, \mathbf{y}, \mathbf{x}^*) = \int_{f^*} p(y^*|f^*)p(f^*|\mathbf{X}, \mathbf{y}, \mathbf{x}^*)df^*, \quad (3)$$

This distribution is typically not computable analytically. However, if the posterior distribution for  $f^*$  is Gaussian (e.g., because a Gaussian likelihood or Gaussian approximate likelihood was used), this integral can often be computed efficiently.

### 2.1 BAYESIAN ACTIVE LEARNING

The goal of Bayesian active learning is to sequentially choose samples so as to accurately model an unknown function  $g(\cdot)$  with as few samples as possible. In the audiometric setting,  $g(\mathbf{x})$  is the probability that the patient hears the tone  $\mathbf{x}$ . If we query whether the patient can hear a set of tones  $\mathbf{X}$ , we would like for our predictive posterior belief  $p(y^*|\mathbf{X}, \mathbf{y}, \mathbf{x}^*)$  to match  $g(\mathbf{x}^*)$  as well as possible and as *confidently* as possible. Suppose that at iteration  $t < n$  the points  $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_t]$  and corresponding labels  $\mathbf{y}$  are

known. Houlshby et al. (2011) propose to use mutual information,

$$I(\mathbf{f}, y_t | \mathbf{x}_t) = H[\mathbf{f} | \mathbf{X}, \mathbf{y}] - \mathbb{E}[H[\mathbf{f} | \mathbf{X}, \mathbf{y}, y_t]]_{p(y_t | \mathbf{x}, \mathbf{y}, \mathbf{x}_t)} \quad (4)$$

where  $H[A]$  denotes the differential entropy of a random variable  $A$ , to identify a new point  $\mathbf{x}_t$ , with future label  $y_t$ , to be queried in iteration  $t$ —*i.e.*  $\mathbf{x}_t$  is chosen to be

$$\mathbf{x}_t = \arg \max_{\mathbf{x}} I(\mathbf{f}, y | \mathbf{x}) \quad (5)$$

### 3 METHOD

In this section, we discuss our model and approach to psychophysical detection testing using Gaussian processes. As a running example, we will use audiometry. In an audiometric detection test, a patient is presented with tones of varying frequency and intensity. The patient is asked to respond (*e.g.*, by pressing a button) if he/she hears the sound. In the absence of a timely reaction the tone is assumed to be inaudible to the patient. The delay between tones is sufficiently randomized to prevent patients from responding to predictable patterns (Gosztonyi Jr et al., 1971).

At time step  $t$ , we choose a tone  $\mathbf{x}_t = (\omega, i)$  with frequency  $\omega$  and intensity  $i$  to present to the subject. In return, we receive a response  $y_t \in \{0, 1\}$ , where  $y_t = 1$  indicates that the patient heard the sound and  $y_t = 0$  indicates that he/she did not. There is inherent observation noise in patient responses. When patients become uncertain when presented with sounds very close to their threshold (*i.e.*, the sounds become faint and hard to hear). Patients do not have perfect detection boundaries, and only hear tones near their hearing threshold with some probability. This uncertainty is observed in reality for a number of reasons. First, patient attention may waver, or they may be unable to distinguish between tones near their hearing threshold and slight background noise. Alternatively, this uncertainty may derive from physical sources. For example, if a tone is faint enough, a patient may be able to hear that tone between—but not during—heart beats. Our goal is therefore to predict the probability that a patient is able hear a given sound.

#### 3.1 PRIOR

In the case of audiometric testing, we have valuable prior knowledge about a patient’s audiometric function that we can encode in our GP model. In particular, the probability that a patient hears a sound  $(\omega, i)$  is *monotonically increasing* in the intensity  $i$ . In other words, if a tone is audible to a patient, then an even louder tone is more likely to be audible. Furthermore, audition is a *smooth function* with respect to the frequency  $\omega$ . Human nerves that detect similar frequencies are co-located in the cochlea and, as a result, a partial loss of hearing in one frequency is likely to cause a loss of hearing in nearby frequencies. A GP prior

can encode both properties naturally through its covariance function. A combination of a linear kernel in intensity and a squared exponential kernel in frequency ensures the monotonicity and smoothness properties:

$$k((\omega, i), (\omega', i')) = ii' + \exp\left\{-\frac{1}{\ell} \|\omega - \omega'\|_2^2\right\}. \quad (6)$$

Here,  $\ell$  regulates the smoothness (characteristic length-scale) w.r.t. frequency. Note that a GP prior is technically incapable of supporting only monotonically increasing functions. However, we only need that the posterior probability of detection,  $3$ , be monotonic, which is generally true after a few tones are sampled (for example, see figure 3).

For the mean function  $\mu_0$ , we note that intensity is typically measured in dB HL, which is an empirical unit of measurement normalized based on population data so that at each frequency the typical human hearing threshold is around 0 dB HL. As a result we choose a constant mean function.

#### 3.2 OBSERVATION MODEL

This mean function,  $\mu_0(\cdot)$ , and covariance function,  $k(\cdot, \cdot)$ , define a prior over real-valued latent functions  $f \sim \mathcal{GP}(\mu_0(\cdot), k(\cdot, \cdot))$ . Our goal is to predict the *probability* (*i.e.* within  $[0, 1]$ ) that a patient hears a tone with a specified frequency and intensity. We can never observe these probabilities directly. For any tone, we can instead only observe the outcome of a Bernoulli trial with the true probability. This setting is akin to Gaussian Process classification (Kuss and Rasmussen, 2005) and similarly we use a Bernoulli likelihood, where  $\Pr(y = 1 | f) = \Phi(f)$  and  $\Phi(\cdot)$  denotes the standard normal cumulative density function (CDF).

The linear component of the kernel in (6) results in a function that, after being warped by  $\Phi(\cdot)$ , is sigmoidal in the intensity dimension: after the slope is fixed (by conditioning on the first few points), the posterior belief about  $\Phi(f)$  will tend to 0 as the intensity decreases and 1 as the intensity increases. This reflects our prior knowledge that tones of extremely low intensity are unlikely to be heard, whereas tones of high intensity are more likely to be audible.

**Predictions.** Once we have collected data, we can use the predictive distribution  $p(y^* | \mathbf{X}, \mathbf{y}, \mathbf{x}^*)$  to summarize our belief about whether the patient will hear a test tone  $\mathbf{x}^*$ . As our likelihood is non-Gaussian, the posterior  $p(f^* | \mathbf{X}, \mathbf{y}, \mathbf{x}^*)$  has no closed form solution. However, an approximate Gaussian posterior over  $f^*$  can be obtained with the standard Laplace approximation to the likelihood (Kuss and Rasmussen, 2005; Rasmussen and Williams, 2006).

### 3.3 MULTIPLE TONES

An interesting property of audiometry (that may also be common to other psychophysical domains, *e.g.* visual or touch sensory tests), is that multiple tone stimuli can be presented to a patient simultaneously by overlaying tones. In this setting however, we can still only query whether the patient heard the overlaid tones. A negative response to a multi-tone sample indicates that the patient did not hear any of the overlaid tones; a positive response indicates that the patient heard *at least one* of them.

**OR-Channel.** Presenting a patient with  $k$  tones leads to a novel extension to the standard Bernoulli likelihood used in classification. We present the patient with  $k$  tones  $\mathbf{x}_1, \dots, \mathbf{x}_k$ . The patient hearing the individual tone  $\mathbf{x}_i$  is still the outcome of a Bernoulli trial with  $\Pr(y_i|f_i) = \Phi(f_i)$ , as the individual trials are independent *conditioned* on  $f$ . However, we cannot directly observe any individual  $y_i$ . Rather, we record them through an *OR-channel*, that is we observe  $\bar{y}$ , which is 1 if the patient hears *at least one* of the  $k$  tones presented, and is 0 otherwise. This leads to the *OR-channel likelihood*:

$$\begin{aligned} \Pr(\bar{y} = 1 | \mathbf{f}_{1..k}) &= 1 - \prod_j (1 - \Phi(f_j)) \\ &= 1 - \prod_j \Phi(-f_j) \end{aligned} \quad (7)$$

Note when  $k=1$ , eq. (7) reduces to the standard Bernoulli likelihood for single tones,  $\Pr(\bar{y} = 1 | f_1) = \Phi(f_1)$ .

### 3.4 QUERY SELECTION

In iteration  $t$  we present the subject with a query set of overlaid tones  $\mathbf{q}_t = [\{\mathbf{x}_1, \dots, \mathbf{x}_k\}]$  and query the response  $\bar{y}_t$ . To select  $\mathbf{q}_t$  we pick the point set that maximizes the expected decrease in posterior entropy, analogous to eq. (4).

**Single tone mutual information.** We first consider the setting of picking a single tone, *i.e.* where  $\mathbf{q}_t = [\{\mathbf{x}_t\}]$ . Houlshy et al. (2011) derive an analytical approximation to the mutual information, eq. (5), when using a Bernoulli likelihood. These results directly apply when picking a single tone  $\mathbf{x}_t$ . When  $f_t$  is known, the entropy of the Bernoulli variable  $y_t$  is given by  $h(\Phi(f_t))$ , where

$$h(p) = -p \log p - (1-p) \log(1-p),$$

is the Bernoulli entropy function. We can rephrase the entropy in eq. (4) as

$$I(\mathbf{f}, y_t | \mathbf{q}_t) = H[y_t | \mathbf{X}, \mathbf{y}] - \mathbb{E}[H[y_t | \mathbf{f}]]_{p(\mathbf{f} | \mathbf{X}, \mathbf{y})}, \quad (8)$$

and rewrite both terms on the right hand side through  $h$ . If  $\mathbf{f}$  is unknown and  $y_t$  is conditioned on  $\mathbf{X}, \mathbf{y}$ , the entropy can

be expressed in terms of the expectation over the posterior for  $f_t$ :

$$H[y_t | \mathbf{X}, \mathbf{y}] = h(\mathbb{E}[\Phi(f_t)]). \quad (9)$$

If  $f_t$  is known we have  $\Pr(y|\mathbf{f}) = \Phi(f_t)$ , yielding

$$H[y_t | \mathbf{f}] = h(\Phi(f_t)). \quad (10)$$

Substituting eqs. (9), (10) into (8) leads us to the following expression for the mutual information between  $\mathbf{f}$  and  $y_t$  in the single tone scenario:

$$I_1(\mathbf{f}, y_t | \mathbf{q}_t) = h(\mathbb{E}[\Phi(f_t)]) - \mathbb{E}[h(\Phi(f_t))]. \quad (11)$$

The computation of  $I_1$  involves an intractable integral, which can be approximated through numerical integration. This approach is very fast in practice as the integral is only one dimensional and can be computed efficiently using quadrature.

**Multiple tone mutual information** The above results can be extended to compute the mutual information when sampling multiple tones  $\mathbf{q}_t = [\{\mathbf{x}_1, \dots, \mathbf{x}_k\}]$ . In particular, the probability of observing  $\bar{y}_t = 1$  changes from  $\Phi(f_t)$  to the OR-channel probability, (7). Thus, when  $f_1, \dots, f_k$  are known, the entropy of the Bernoulli variable  $\bar{y}_t$  is  $h\left(1 - \prod_{i=1}^k \Phi(-f_i)\right)$ .

To simplify notation, let us define  $\bar{p}_1 = \Pr(\bar{y} = 1 | \mathbf{f}_{1..k})$  as defined in (7). Substituting  $\bar{p}_1$  for  $\Phi(f_t)$  in (11) gives the mutual information of paired tone sample  $\mathbf{q}_t$  after observing the outcome  $\bar{y}_t$ :

$$\begin{aligned} I_k(\mathbf{f}, \bar{y}_t | \mathbf{q}_t) &= h(\mathbb{E}[\bar{p}_1]) - \mathbb{E}[h(\bar{p}_1)] \\ &= h\left(\mathbb{E}\left[\prod_j \Phi(-f_j)\right]\right) - \mathbb{E}\left[h\left(\prod_j \Phi(-f_j)\right)\right] \end{aligned} \quad (12)$$

where the second equality holds by the linearity of expectation and because  $h(p)$  is a concave function that is symmetric about  $p=0.5$  (*i.e.*  $h(p) = 1 - h(p)$ ). The last term leads again to an intractable integral. However, similar to the one tone scenario,  $I_k$  can also be evaluated efficiently using numerical integration, as  $k$  is relatively small.

**Computational Considerations** Finding a set of  $k \leq K$  tones  $\mathbf{q}_t^{(k)}$  to maximize  $I_k(\mathbf{f}, \bar{y}_t | \mathbf{q}_t)$  from a candidate set  $\mathcal{X}$  of size  $S$  requires  $O\left(\binom{S}{k}\right)$  considerations. In order to ensure that patients do not have to wait for a lengthy duration between sounds are played, we construct a set of multiple tones to play greedily. We select the best single tone by exhaustively searching  $\mathcal{X}$ . Then, to select the best set of size  $k$ , we exhaustively add each  $\hat{\mathbf{x}} \in \mathcal{X}$  to the best set of size  $k-1$ ,  $\mathbf{q}_t^{(k-1)}$  and compute the expected decrease in posterior entropy of  $\mathbf{q}_t^{(k-1)} \cup \hat{\mathbf{x}}$ . This greedy selection procedure reduces the computational complexity of considering tone sets of up to size  $k$  to  $O(Sk)$ , and in practice requires only a few seconds of computation time.

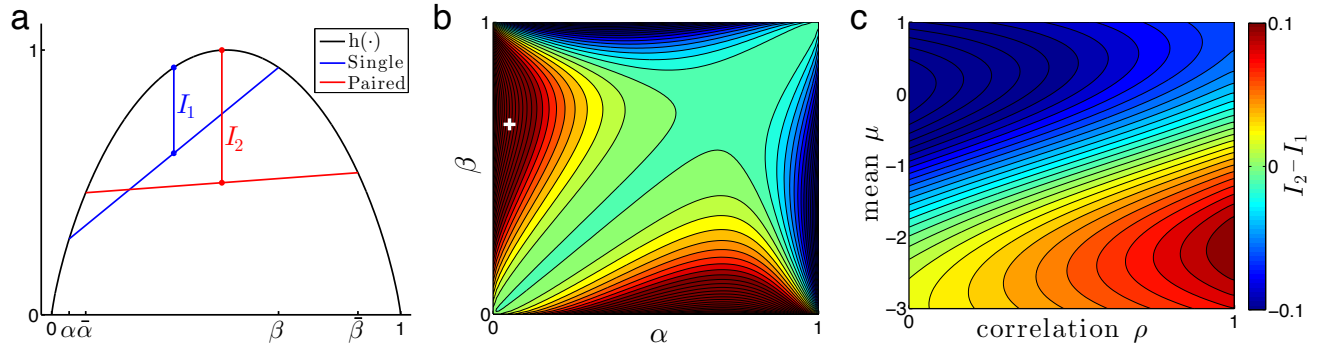


Figure 1: Difference in mutual information  $I_2 - I_1$  between a paired query and a single query: **(a)** discrete distribution with two atoms  $(\alpha, \beta) = 0.05, 0.65$ , and corresponding  $\bar{\alpha} = 1 - (1 - \alpha)^2 \approx 0.1$ ,  $\bar{\beta} \approx 0.88$ ). Here  $I_2 - I_1 \approx 0.18$ ; **(b)**  $I_2 - I_1$  as a function of  $\alpha, \beta$  (white cross denotes the specific example of panel a); **(c)** the normally distributed latent input case.  $I_2 - I_1$  is shown as a function of the mean  $\mu$  and correlation  $\rho$ . Colorbar at right is for both panel b and c.

### 3.5 OR-CHANNEL ANALYSIS

We first investigate the OR-channel likelihood of eq. (7), as it is unclear if this elaboration can provide any benefit over a standard Bernoulli likelihood. Intuitively, the result of  $\bar{y} = 0$  from an OR-channel is quite informative: all inputs into that channel must have been 0 (in the auditory example, no sounds were heard). On the other hand, the result of  $\bar{y} = 1$  is much less informative than in the Bernoulli channel, as it means only that one or more of the inputs were 1 (some sound or sounds were heard), but there is no information about which. Here we analyze simple models that support the use of the OR-channel likelihood. We compare a *single* input, corresponding to the standard Bernoulli likelihood, to a *paired* input, corresponding to an OR-channel likelihood with two inputs. That is, with inputs  $\{f_1, f_2\}$  and output  $y \in \{0, 1\}$  as above, our quantities of interest are  $I_1 := I(y, f_1)$  and  $I_2 := I(y, \{f_1, f_2\})$ , and we seek to understand if more information about the inputs can exist in the paired-input query, than in the single-input query.

#### 3.5.1 OR-channel Inputs With Discrete Support

The simplest case involves perfectly correlated inputs  $f_1 = f_2$ , and further, a discrete distribution on  $f_1$  with two atoms of equal mass. The implied probability  $\phi(f_1)$  will then have the same discrete distribution, which we write as  $p(\phi(f_1)) = \frac{1}{2}\delta(\phi(f_1) = \alpha) + \frac{1}{2}\delta(\phi(f_1) = \beta)$ , for some atoms  $\alpha$  and  $\beta$ . Then, the mutual information of the single query is:

$$\begin{aligned} I_1 &= H(y) - H(y|f_1) \\ &= h(\mathbb{E}_f[\phi(f_1)]) - \mathbb{E}_f[h(\phi(f_1))] \\ &= h\left(\frac{1}{2}(\alpha + \beta)\right) - \frac{1}{2}(h(\alpha) + h(\beta)), \end{aligned} \quad (13)$$

where  $\mathbb{E}_f$  is the expectation under the distribution on  $f$ . The OR-channel likelihood for two terms is similarly

$p(y = 1|\{f_1, f_2\}) = 1 - (1 - \phi(f_1))(1 - \phi(f_2)) = 1 - (1 - \phi(f_1))^2$ . The mutual information of a paired-input query becomes

$$I_2 = h\left(\frac{1}{2}(\bar{\alpha} + \bar{\beta})\right) - \frac{1}{2}(h(\bar{\alpha}) + h(\bar{\beta})), \quad (14)$$

where  $\bar{\alpha} = 1 - (1 - \alpha)^2$  and  $\bar{\beta} = 1 - (1 - \beta)^2$ .  $I_2$  and  $I_1$  offer a convenient geometric interpretation by viewing mutual information as the Jensen's inequality gap of  $h$  (eqs. (13) and (14)). With this simple discrete distribution,  $\alpha$  and  $\beta$  can be chosen such that  $I_2 - I_1$  will be positive or negative. We show the critical case  $I_2 > I_1$  in Figure 1a, where the blue line segment connects  $(\alpha, h(\alpha))$  to  $(\beta, h(\beta))$  with  $(\alpha, \beta) = (0.05, 0.65)$ , and the red line segment is then implied by those choices of  $\alpha, \beta$  (that is,  $(\bar{\alpha}, \bar{\beta}) \approx (0.10, 0.88)$  in the figure). Here the difference is  $I_2 - I_1 = 0.18$  bits. The contours of  $I_2 - I_1$  as a function of  $(\alpha, \beta)$  is shown in Figure 1b.

#### 3.5.2 OR-channel Inputs With Normal Densities

We next analyze the OR-channel likelihood with two latent factors  $f_1 = f(x_1)$  and  $f_2 = f(x_2)$ , which are jointly Gaussian according to the GP model of Section 3:  $[f_1, f_2] \sim \mathcal{N}(m, S)$ . We calculate  $I_2 - I_1$  numerically using eq. (11) (note that, compared to the previous example, only the expectation over  $f$  has changed). We simplify the parameter space with  $m = \begin{bmatrix} \mu \\ \mu \end{bmatrix}$  and  $S = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}$  (but note that the function  $I_2 - I_1$  is not invariant to either of these simplifications). We plot the contours of  $I_2 - I_1$  as a function of correlation  $\rho$  and mean  $\mu$  in Figure 1c, which indeed has substantial regions of both positive and negative mass.

In summary, though intuitively non-obvious, the above analyses clarify that the OR-channel likelihood can, but need not, increase mutual information between the input distribution and the binary outcome  $y$ . This finding offers a

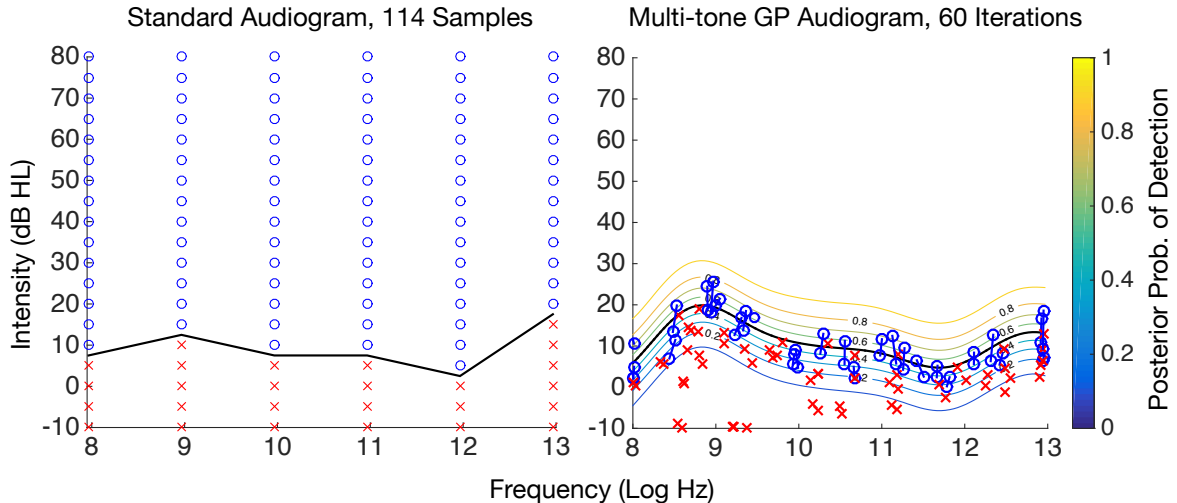


Figure 2: Standard grid search audiogram with tones played at every octave from 250 to 8000 Hz, and every 5 dB HL from -10 dB to 80 dB, compared to a multi-tone GP audiogram with 60 iterations (and therefore 119 “samples”).

critical takeaway: the OR-channel can be used effectively, but only in the setting where a judicious choice of input distribution can be made. Indeed, this is exactly what our framework will achieve: it will choose pairs of input points (paired sounds) to learn more about the underlying audiogram than a single point alone. Thus, the OR-channel likelihood offers benefit beyond this scheme, which we already expect to outperform a naive approach to learning these latent functions. In this work we only consider paired inputs; a future question for study is how the information gain distribution changes with increasing numbers of inputs.

## 4 RELATED WORK

A number of papers have been recently published on Bayesian active learning. Many papers have considered Bayesian active learning using mutual information in the regression setting (Guestrin et al., 2005; Krause and Guestrin, 2007; Srinivas et al., 2009). However, the computation of mutual information is significantly less tractable in the classification setting. To our knowledge, Housby et al. (2011) is the first paper to leverage the rewriting of mutual information in (12), allowing for tractable computation of mutual information with the Bernoulli observation model. This paper is most similar to ours, as the Bernoulli observation model is identical to our single tone audiometric algorithm. A number of other, orthogonal applications and extensions of this method have since been published (Garnett et al., 2013; Iwata et al., 2013).

Alternative techniques for estimating audiograms have existed for many years. Sweep-based audiometry, such as Bekesy audiometry and Audioscan, are able to produce a more continuous estimate of the audiogram that can often detect notches, but with the disadvantage of a partic-

ularly time- and attention-demanding task (Jerger, 1960; Meyer-Bisch, 1996). Several Bayesian audiogram estimation techniques, such as parameter estimation by sequential testing (PEST) and maximum likelihood methods also exist, although most do not simultaneously estimate multiple frequencies (Green, 1993; Leek et al., 2000; Özdamar et al., 1990; Pentland, 1980; Taylor and Creelman, 1967). More recent advances in audiometric testing have focused on improving the accessibility of hearing screening by distribution over telephone, Internet, or mobile devices (Smits et al., 2004; Swanepoel et al., 2014; Vlaming et al., 2014; Watson et al., 2012; Williams-Sanchez et al., 2014).

## 5 RESULTS

In this section, we empirically evaluate our proposed algorithms for psychophysical detection. We focus on our application to audiometry, and seek to evaluate the merits of using Gaussian processes for audiometry in general, as well as to compare single-tone and multi-tone audiometry, focusing on the machine learning aspects of our algorithms.

We have since published a small clinical trial in a medical journal evaluating the novel GP audiometric techniques discussed here from a clinical point of view as well, and refer readers to Song et al. (2015) for additional results comparing GP audiometry and standard audiometry.

To begin, we compare the audiograms found by a standard grid audiometric test and by our multi-tone GP model. In both cases we run the same human subject in the same audiometric setting. The only differences are the tones presented and the method used to infer the audiometric function. All audiometric tests were run in accordance with an approved IRB. In the standard setting, tones from this

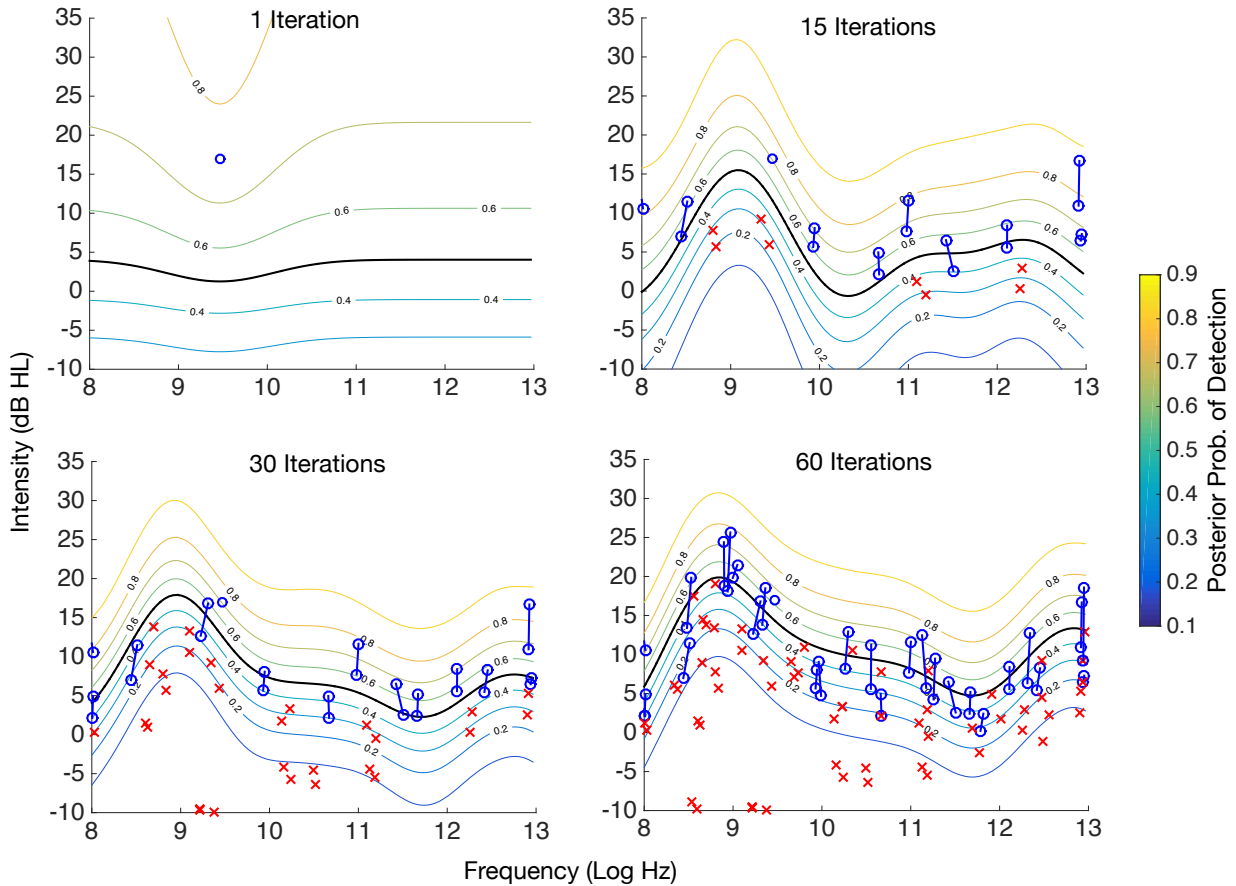


Figure 3: The posterior probability of detection within the frequency / intensity space during a GP audiometric test on a human subject. Panels show the learned GP after 1, 15, 30, and 60 iterations. Queries consist of a single or a paired tone (as selected by the model). Blue circles indicate a positive outcome (sound was heard), red crosses indicate a negative outcome. Paired tones with positive outcome (at least one of the two tones was heard) are connected by a blue line. Almost all queries are close to the final audible threshold (0.5 posterior detection probability), which is well approximated even after only 15 iterations.

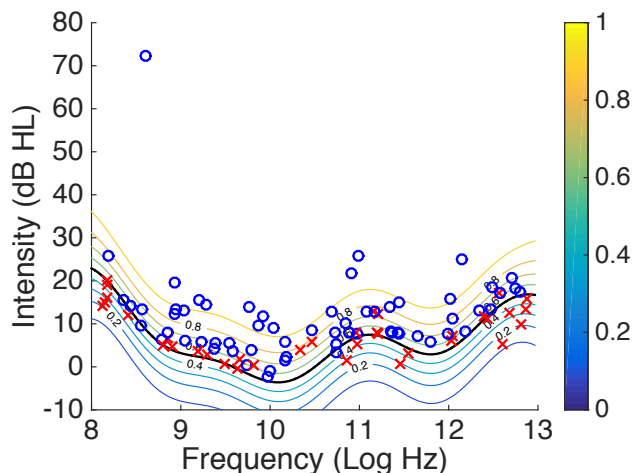
grid are presented in a pre-determined order, typically ascending in frequency and decreasing in intensity. In the GP model, pairs of tones were actively selected given all previous pairs of tones and the responses to those tones. A random delay of up to 3 seconds was inserted between tone presentations to prevent subjects from memorizing a pattern in the test. Figure 2 shows the resulting data and inferred audiograms plotted in frequency-intensity space (left panel: standard audiometric test; right panel: GP method).

For both the standard and GP experiments, tones that were detected by the patient are plotted as blue circles, and tones that were not detected are plotted as red crosses. For the paired-tone GP test (right panel), paired samples that were detected are plotted as blue circles connected by a blue line (recall that, due to the OR-channel likelihood, we do not know which tone was heard). Paired tones that were not detected are again plotted as individual red crosses, as these data are functionally equivalent to two single-tone samples

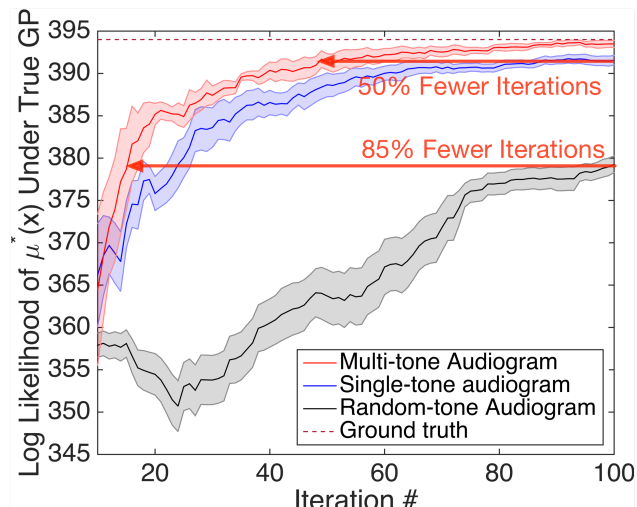
that were not detected (again due to the OR-channel observation model).

In the standard audiometric test, the inferred audiogram is simply an “audible threshold” that is the piecewise linear function connecting the detection threshold at each frequency. This threshold is depicted as a black line in the left panel of Figure 2. In the GP case, we infer a full posterior distribution on the detection threshold. We plot contours of the posterior detection probability in Figure 2, with a solid black line at 50% posterior detection probability.

This confirmatory comparison offers several key points of interpretation. First, the tests agree with each other: the 50% posterior detection probability in the GP case is within 5dB of the standard audiogram, giving confidence to the general sensibility of this model. Second, perhaps most importantly to the active learning goal, the GP active learning



(a) A GP trained on 100 single tones. Blue circles denote tones detected by the subject, and red crosses denote tones that were not detected. The posterior probabilities are shown as color contours.



(b) Log likelihood of random presentation of tones (no active learning, shown in gray), active learning presentation of single tones (shown in blue), and active learning with paired tones (shown in red), under the ground truth audiometric function from Figure 4a. Log likelihood is plotted as a function of iterations in each audiometric testing strategy. Shaded areas denote standard error.

Figure 4: Comparison of multi-tone and single-tone GP audiometrics

model presents approximately half as many iterations (60 actively learned paired tones compared to 114 single tones preselected from a grid). Thus the GP model is able to explore substantially more of the frequency space than the standard grid test, and it does so in many fewer overall iterations, reducing the burden of these tests. Third, note that the GP model does not explore uninformative regions of tone space: above a certain intensity (at which the model is confident that tones are certainly heard), there are no tones queried. This observation differs sharply from the standard test, which squanders numerous samples at intensities well above this subject’s audible threshold, where little to no information is available. Fourth, by design our GP model offers a full posterior distribution over tone space, and thus produces a richer and more descriptive audiogram than the piecewise linear audible threshold function in the standard test. Finally, it is worth noting that, though the paired tones in the right panel of Figure 2 appear to be sampled at very similar frequencies in log-space, the differences were often nontrivial, up to four or five half steps in an octave.

Next, Figure 3 investigates the convergence of our GP model after 1, 15, 30, 60 iterations of our paired-tone GP audiometric algorithm. The posterior after a single iteration (upper left panel) reflects primarily the prior mean and the covariance of the model, which incorporates our knowledge about the general shape of human audiograms. As the active learning procedure continues (other panels), the GP posterior quickly converges to the audiogram of this particular subject. After only 30 iterations, the GP model

has already captured the audiogram shape, and subsequent changes are very minor.

To investigate the performance of our GP active learning method in greater detail, we construct a synthetic data set with known ground truth (a known audiometric function). We begin by training a GP on 100 single tones and the detection of those tones reported by a second human subject. The tones sampled and the inferred audiogram are presented in Figure 4a. We use this posterior GP as the true audiogram of a simulated subject.

This ground truth audiometric function allows for the critical assessment of performance shown in Figure 4b. We compare three strategies of data presentation: random presentation of tones (no active learning, shown in gray), active learning presentation of single tones (shown in blue), and active learning with paired tones (shown in red). For each strategy, at each iteration (tone presentation), we infer the GP posterior mean, which is the MAP estimate of the audiometric function, given each stream of data. We evaluate the log likelihood of each strategy’s GP posterior mean under the ground truth GP from Figure 4a. This step offers a quantitative assessment of how closely each strategy has approximated the true audiometric function. The maroon dashed line depicts the log likelihood of the ground truth GP itself, which is thus the maximum achievable performance of any strategy. All three strategies (random, single tone active learning, paired tone active learning) should, with enough iterations, converge to ground truth. Thus, the essential question of this work, and indeed of any active

learning method, is how much more quickly a particular strategy approaches the ground truth than competing strategies.

We ran the single and paired tone active learning methods ten times each, and standard errors are plotted as shaded regions. Because of the very high standard error of the random tone audiogram, these results were averaged over 100 runs.

Figure 4b has a few key findings. Both the single and paired tone active learning strategies significantly outperform random sampling. Thus our strong prior rapidly learns that large portions of the tone space are either very likely or very unlikely to be heard, and is able to quickly learn to sample in regions of high information. After 80-90 iterations the paired tone algorithm matches the ground truth model very closely. This result is in significant contrast to randomly choosing tones, which not only has very large standard error, but also rarely converges to a good model. Finally, we observe that the paired tone active learning strategy significantly outperforms the single tone strategy. In fact, the paired tone strategy requires only half as many iterations to achieve the same level of likelihood. Compared to random sampling, paired tone active learning reduces the number of iterations by 85%.

## 6 DISCUSSION

In this paper, we explored the problem of adapting Bayesian active learning to psychophysical testing, and improving upon standard techniques used in audiometric testing. In the process of our investigation, we developed a novel OR-channel likelihood that allows us to present multiple tones to a subject simultaneously, leading to an audiometric testing strategy that not only yields good audiogram estimation using significantly fewer samples, but also leads to much better coverage of the frequency dimension. We demonstrate a non-obvious result, that multiple tones played through an OR-channel can, but do not have to, yield more information than a single tone. As future work we will continue to investigate the theoretical properties of this likelihood function and its use in active learning. We also hope that the drastic improvements of our method over the state-of-the-art will convince experts in medicine and psychology to adapt machine learned approaches for psychophysical testing.

## 7 ACKNOWLEDGEMENTS

KQW and JRG are supported by NIH grant U011U01NS073457-01 and NSF grants IIA-1355406, IIS-1149882, EFRI-1137211, CNS-1017701, CCF-1215302, and IIS-1343896. XS and DB are supported by NIH grant R01-DC009215. JPC is supported by a Sloan Research Fellowship.

## References

- Raymond Carhart and James Jerger. Preferred method for clinical determination of pure-tone thresholds. *Journal of Speech & Hearing Disorders*, 1959.
- Matt Carter and Jennifer C Shieh. *Guide to research techniques in neuroscience*. Academic Press, 2009.
- Manuel Don, Jos J Eggermont, and Derald E Brackmann. Reconstruction of the audiogram using brain stem responses and high-pass noise masking. *The Annals of otology, rhinology & laryngology. Supplement*, (3 Pt 2 Suppl 57):1–20, 1978.
- Roman Garnett, Michael A Osborne, and Philipp Hennig. Active learning of linear embeddings for gaussian processes. *arXiv preprint arXiv:1310.6740*, 2013.
- Rudolph E Gosztonyi Jr, Lawrence A Vassallo, and Joseph Sataloff. Audiometric reliability in industry. *Archives of Environmental Health: An International Journal*, 22(1): 113–118, 1971.
- David M Green. A maximum-likelihood method for estimating thresholds in a yes–no task. *The Journal of the Acoustical Society of America*, 93(4):2096–2105, 1993.
- Carlos Guestrin, Andreas Krause, and Ajit Paul Singh. Near-optimal sensor placements in gaussian processes. In *ICML*, 2005.
- Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*, 2011.
- WAITER Hughson and Harold Westlake. Manual for program outline for rehabilitation of aural casualties both military and civilian. *Trans Am Acad Ophthalmol Otolaryngol*, 48(Suppl):1–15, 1944.
- Tomoharu Iwata, Neil Houlsby, and Zoubin Ghahramani. Active learning for interactive visualization. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, pages 342–350, 2013.
- James Jerger. Bekesy audiometry in analysis of auditory disorders. *Journal of Speech, Language, and Hearing Research*, 3(3):275–287, 1960.
- Andreas Krause and Carlos Guestrin. Nonmyopic active learning of gaussian processes: an exploration-exploitation approach. In *ICML 24*, 2007.
- Malte Kuss and Carl Edward Rasmussen. Assessing approximate inference for binary gaussian process classification. *The Journal of Machine Learning Research*, 6: 1679–1704, 2005.
- Marjorie R Leek, Judy R Dubno, Ning-ji He, and Jayne B Ahlstrom. Experience with a yes–no single-interval maximum-likelihood procedure. *The Journal of the Acoustical Society of America*, 107(5):2674–2684, 2000.



- Ted Madison et al. Guidelines for manual pure-tone threshold audiometry. 2005.
- Christian Meyer-Bisch. Audioscan: a high-definition audiometry technique based on constant-level frequency sweeps—a new method with new hearing indicators. *International Journal of Audiology*, 35(2):63–72, 1996.
- Thomas P Minka. Expectation propagation for approximate bayesian inference. In *UAI*, 2001.
- Özcan Özdamar, Rebecca E Eilers, Edward Miskiel, and Judith Widen. Classification of audiograms by sequential testing using a dynamic bayesian procedure. *The Journal of the Acoustical Society of America*, 88(5):2171–2179, 1990.
- Alex Pentland. Maximum likelihood estimation: The best pest. *Attention, Perception, & Psychophysics*, 28(4):377–379, 1980.
- C.E. Rasmussen and C.K.I. Williams. Gaussian processes for machine learning. MIT Press, 2006.
- DW Robinson. Long-term repeatability of the pure-tone hearing threshold and its relation to noise exposure. *British journal of audiology*, 25(4):219–235, 1991.
- U Schiefer, J Pätzold, and F Dannheim. Konventionelle perimetrie. *Der Ophthalmologe*, 102(6):627–646, 2005.
- Nicolas Schmuziger, Rudolf Probst, and Jacek Smurzynski. Test-retest reliability of pure-tone thresholds from 0.5 to 16 khz using sennheiser hda 200 and etymotic research er-2 earphones. *Ear and hearing*, 25(2):127–132, 2004.
- Cas Smits, Theo S Kapteyn, and Tammo Houtgast. Development and validation of an automatic speech-in-noise screening test by telephone. *International journal of audiology*, 43(1):15–28, 2004.
- X. D. Song, B. M. Wallace, J. R. Gardner, N. M. Ledbetter, K. Q. Weinberger, and D. L. Barbour. Fast, continuous audiogram estimation using machine learning. *Ear and Hearing*, 2015.
- Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*, 2009.
- De Wet Swanepoel, Hermanus C Myburgh, David M Howe, Faheema Mahomed, and Robert H Eikelboom. Smartphone hearing screening with integrated quality control and data management. *International journal of audiology*, 53(12):841–849, 2014.
- MiM Taylor and C Douglas Creelman. Pest: Efficient estimates on probability functions. *The Journal of the Acoustical Society of America*, 41(4A):782–787, 1967.
- Marcel SMG Vlaming, Robert C MacKinnon, Marije Jansen, and David R Moore. Automated screening for high-frequency hearing loss. *Ear and hearing*, 35(6):667, 2014.
- Charles S Watson, Gary R Kidd, James D Miller, Cas Smits, and Larry E Humes. Telephone screening tests for functionally impaired hearing: Current use in seven countries and development of a us version. *Journal of the American Academy of Audiology*, 23(10):757–767, 2012.
- Victoria Williams-Sanchez, Rachel A McArdle, Richard H Wilson, Gary R Kidd, Charles S Watson, and Andrea L Bourne. Validation of a screening test of auditory function using the telephone. *Journal of the American Academy of Audiology*, 25(10):937–951, 2014.
- F Zhao, D Stephens, and C Meyer-Bisch. The audioscan: a high frequency resolution audiometric technique and its clinical applications. *Clinical Otolaryngology & Allied Sciences*, 27(1):4–10, 2002.
- Fei Zhao and Dafydd Stephens. Analyses of notches in audioscan and dpoaes in subjects with normal hearing. *International Journal of Audiology*, 37(6):335–343, 1998.

---

# Locally Conditioned Belief Propagation

---

**Thomas Geier** and **Felix Richter** and **Susanne Biundo**  
Institute of Artificial Intelligence  
Ulm University, Germany  
{thomas.geier, felix.richter, susanne.biundo}@uni-ulm.de

## Abstract

Conditioned Belief Propagation (CBP) is an algorithm for approximate inference in probabilistic graphical models. It works by conditioning on a subset of variables and solving the remainder using loopy Belief Propagation. Unfortunately, CBP’s runtime scales exponentially in the number of conditioned variables. Locally Conditioned Belief Propagation (LCBP) approximates the results of CBP by treating conditions locally, and in this way avoids the exponential blow-up. We formulate LCBP as a variational optimization problem and derive a set of update equations that can be used to solve it. We show empirically that LCBP delivers results that are close to those obtained from CBP, while the computational cost scales favorably with problem size.

## 1 INTRODUCTION

Modern SAT solvers are capable of solving problem instances with hundreds of thousands of variables (Katebi et al., 2011), despite the fact that SAT is an NP-hard problem. Most of today’s practical solvers are CDCL (conflict-driven, clause-learning) solvers (Marques-Silva et al., 2009). Their main algorithmic components are branching, unit propagation, and clause learning (Katebi et al., 2011). Generalizing these concepts, we could talk of branching as analysis by cases, unit propagation as inference within a single case (both already found in the classic DPLL algorithm (Davis et al., 1962)), and clause learning (Silva and Sakallah, 1996) as reusing inference results across cases.

The #P-hard (Roth, 1996) problem of computing marginal probabilities (or the partition function) in discrete-valued graphical models is closely related to #SAT—the task of counting the models of a propositional formula. Probabilistic inference generalizes the boolean conjunction of

clauses to a product over local real-valued functions. The #SAT problem is usually tackled using modified CDCL solvers (Bayardo Jr and Pehoushek, 2000; Sang et al., 2004; Huang and Darwiche, 2005). Both for probabilistic inference and #SAT, it is not enough to find one satisfying case, but one has to take into consideration all cases. But while SAT problems are usually sparse, probabilistic problems can often be strictly positive. It is thus not very surprising that the basic probabilistic inference algorithms do not employ analysis by cases, but rely on inference by propagation only: Variable elimination (Koller and Friedman, 2009, Chapter 9), the Junction tree method (Shenoy and Shafer, 1990), loopy Belief Propagation (Pearl, 1986), and more generally the class of algorithms with variational interpretations (Wainwright and Jordan, 2008) can be counted towards this class.

But there are also algorithms that complement propagation with an analysis by cases, such as the exact Recursive Conditioning (Darwiche, 2001) and Value Elimination (Bacchus et al., 2002). Also there exist approximate instances: Cutset Sampling (Bidyuk and Dechter, 2007), SampleSearch (Gogate and Dechter, 2011), Conditioned Belief Propagation (Eaton and Ghahramani, 2009) and collapsed sampling algorithms in general, just to name a few. These approaches appear to have an advantage when the problem encodes a distribution that is not strictly positive, i.e., factors can evaluate to zero (we call these factors deterministic dependencies). Under the presence of deterministic dependencies, analysis by cases is able to reveal context-specific independencies, and prune the search space without incurring an approximation error.

The third algorithmic component in SAT solvers is reusing results across cases. Notably both named exact algorithms heavily rely on this concept under the name of caching. Of the named approximate inference algorithms that employ analysis by cases, only the importance sampler SampleSearch shares work across cases, and only for deterministic dependencies, by using no-good learning (Dechter, 1990). Conditioned Belief Propagation (CBP) (Eaton and Ghahramani, 2009; Geier et al., 2014a) is the straight-

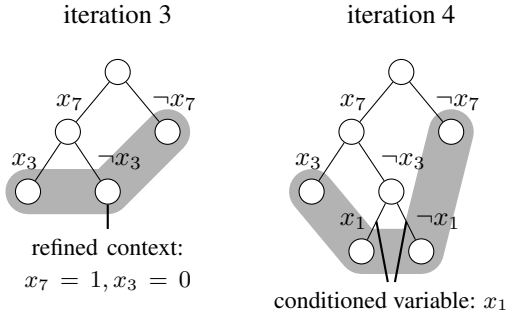


Figure 1: Iterative CBP (Geier et al., 2014a) is a divide-and-conquer algorithm that splits the problem by recursive conditioning. The state of the algorithm is defined by a tree, where edges represent assignments to variables, and nodes represent the partial assignment defined by their path from the root (or the sub-problem obtained by conditioning on this assignment). In each iteration a leaf node (case, context) is chosen and further refined by splitting on an unassigned variable. An approximation to the partition function is obtained by summing the partition function estimates found by BP on each leaf. Marginal probabilities can be obtained by forming a convex combination of the corresponding estimates for the leaves, using their estimated partition functions as weights.

forward combination of systematic analysis by cases (conditioning) with loopy Belief Propagation (BP) as approximate inference within each case. It works by recursively splitting on the assignments to single variables, producing an unbalanced and dynamically ordered tree in the process (Figure 1). This appears to be a fertile combination, as BP yields good results in weakly coupled (high entropy) models and suffers under the presence of strong dependencies (Montanari and Rizzo, 2005; Mooij and Kappen, 2007). Contrarily, conditioning provides benefits for low entropy models with strong dependencies, but fails when the probability mass is spread out evenly over a large number of similar conditions. As shown empirically by Geier et al. (2014b), CBP is indeed able to deliver good improvements over plain BP in particular for low entropy distributions (Figure 2). But the same work also highlights one major shortcoming of CBP: To sustain the same proportional improvement, the number of cases CBP has to evaluate increases exponentially with problem size (Figure 2).

In this essay we describe a method to improve the CBP algorithm in such a way that work between cases is shared approximately—thus adding the third algorithmic component found in modern SAT solvers. The basic idea focuses on the observation that the influence of conditioning on variables usually diminishes with graphical distance. We underpin this assumption empirically by visualizing the effect of conditioning a single variable in randomly generated grid problems in Figure 3. We exploit this “locality of ef-

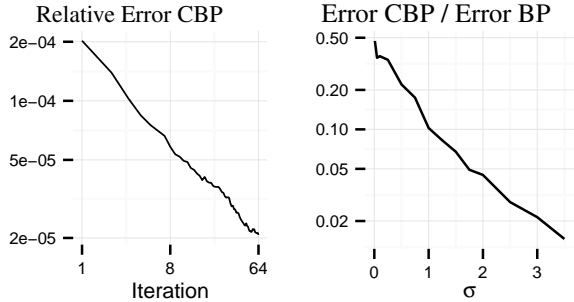


Figure 2: The plots show the typical behavior of iterative CBP on random binary-valued  $8 \times 8$  grid problems (Geier et al., 2014b). The left plot shows the relative error in  $\ln Z$  (median over 500 problems, factor values sampled from  $\exp(\mathcal{N}(0, \sigma))$  with  $\sigma = 1$ ), which *improves only logarithmically* with the number of distinguished cases (Iterations). The right plot shows the CBP error after 64 iterations as a fraction of the BP error for problems with varying strength of interaction (higher  $\sigma$  corresponds to stronger interactions,  $\sigma = 0$  excluded, median over 250 problems). The approximation error of CBP compared to the error of BP consistently decreases with stronger dependencies.

fect” assumption in the proposed *Locally Conditioned Belief Propagation* (LCBP) model. LCBP conceptually works by merging nodes of the BP graph between different cases of CBP, thus effectively sharing message values. An intuition of the difference between CBP and LCBP is conveyed by Figure 4.

## 2 PRELIMINARIES

We focus on undirected graphical models over  $n$  random variables  $X_1, X_2, \dots, X_n$ , referring to the set of all variables as  $\mathcal{X}$ . Each variable  $X_i \in \mathcal{X}$  may assume values out of its finite domain  $\text{Dom}(X_i)$ . A problem is given by a finite set  $\Phi$  of non-negative local functions (factors). Each function  $\phi_a \in \Phi$  is defined over the valuations  $\text{Val}(\mathbf{X}_a)$  (assignments of values to variables) for a subset  $\mathbf{X}_a \subseteq \mathcal{X}$  of variables. The (unnormalized) product over all factors is  $\tilde{p}(x) = \prod_a \phi_a(x_a)$ , and it implies a proper distribution by

$$p(x) = \frac{1}{Z} \tilde{p}(x) \text{ with } Z = \sum_x \tilde{p}(x). \quad (1)$$

The normalizing constant  $Z$  is called the *partition function*.

### 2.1 BELIEF PROPAGATION AS OPTIMIZATION

Given a factorized distribution  $p$ , the basic problem of probabilistic inference is to compute some property of it. These properties are usually expectations, marginal probabilities, the partition function, or most probable

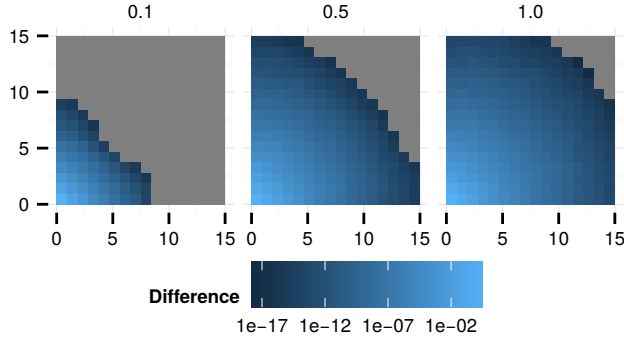


Figure 3: Comparison between two runs of BP on a  $16 \times 16$  grid problem of binary-valued variables. The color encodes the difference of the marginal probabilities after conditioning the variable in the lower left (median over 100 random instances). Factor values are drawn from an exponentiated normal distribution  $\exp(\mathcal{N}(0, \sigma))$  with standard deviation  $\sigma \in \{0.1, 0.5, 1\}$ . Gray means the difference is lower than numerical accuracy. One can see that the effect of the conditioning is local to the conditioned variable. The range of the effect increases with stronger potentials.

assignments—and their exact computation is often intractable (Roth, 1996). Variational inference (Wainwright and Jordan, 2008) is a form of approximate inference that works by substituting  $p$  by some element  $q$  from a class of (pseudo-) distributions  $\mathcal{Q}$ , on the members of which inference is tractable. The instance  $q$  is chosen to be as close to  $p$  as possible. The notion of closeness is captured by some distance measure, which is often taken to be the Kullback-Leibler divergence—though other measures are possible (Minka, 2005).

From the KL-divergence between  $q$  and  $p$  one can obtain

$$\ln Z = \mathbb{E}_q[\ln \tilde{p}] + \mathbb{H}(q) + \text{KL}(q \parallel p). \quad (2)$$

Here,  $\mathbb{H}(q)$  denotes the entropy of  $q$ , and  $\mathbb{E}_q[f(x)]$  denotes the expectation of  $f(x)$  taken with respect to the measure  $q$ . From Equation 2 we identify the functional  $F(q)$ , known as the negative free energy:

$$F(q) = \mathbb{E}_q[\ln \tilde{p}] + \mathbb{H}(q) \quad (3)$$

It yields the exact log-partition function if  $q = p$ , and can serve as a lower bound to  $Z$  if the class  $\mathcal{Q}$  contains only valid distributions. The task in variational inference is to find a  $q^*$  that maximizes  $F$ . In general, either because there exists no exact representation of  $p$  in  $\mathcal{Q}$ , because the class  $\mathcal{Q}$  also contains non-distribution functions, or because we cannot solve the optimization problem perfectly, the found value of  $F(q)$  can only serve as an approximation for  $\ln Z$ . In addition, for many interesting classes  $\mathcal{Q}$ , the functional cannot be given in closed form and one has to resort to further approximations.

We briefly summarize how to express the BP algorithm as

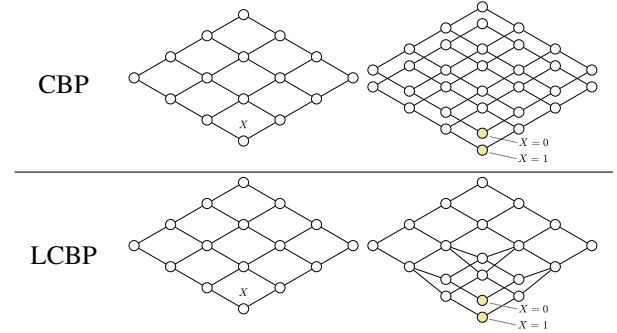


Figure 4: The upper row represents how CBP works by making full copies of the problem for each case. The lower row shows how LCBP only makes copies of the nodes that are local to the conditioned variable.

a variational optimization problem. A more detailed exposition can be found in Yedidia et al. (2005) and Koller and Friedman (2009, Chapter 11). For deriving the BP message update equations using the variational approach, members  $q$  of class  $\mathcal{Q}_{\text{BP}}$  are defined by marginal distributions  $q_i(X_i)$  over the variables (called variable beliefs), and marginal distributions  $q_a(\mathbf{X}_a)$  over the factors in  $\Phi$  (called factor beliefs):

$$q(x) = \prod_a q_a(x_a) \prod_i q_i(x_i)^{(1-d_i)} \quad (4)$$

Here,  $d_i = |\{\phi_a \in \Phi \mid X_i \in \mathbf{X}_a\}|$  represents the number of factors that depend on variable  $X_i$ . In addition to being proper probability measures (sum to one, non-negative), the variable and factor beliefs have to be consistent with respect to their marginal probabilities. This is formalized by requiring for all factors  $\phi_a$ , adjacent variables  $X_i \in \mathbf{X}_a$ , and values  $x_i \in \text{Val}(X_i)$ :

$$\sum_{x_a \models x_i} q_a(x_a) = q_i(x_i) \quad (5)$$

Note that we write  $x_a \models x_i$  for all the (partial) assignment  $x_a \in \text{Val}(\mathbf{X}_a)$  that are an extension of  $x_i$ . A further ingredient in the variational derivation of standard BP exists in an approximation to the entropy, known as the Bethe-Peierls (also BP) approximation and given by

$$\mathbb{H}_{\text{BP}}(q) = \sum_a \mathbb{H}(q_a) + \sum_i (1 - d_i) \mathbb{H}(q_i). \quad (6)$$

A justification for  $\mathbb{H}_{\text{BP}}$  is usually given by the fact that it is exact for tree-structured problems. The BP approximation together with the assumption that the functions  $q_a$  resemble marginal distributions of  $q$  over the variables in  $\mathbf{X}_a$  yields the functional

$$F_{\text{BP}}(q) = \sum_a \mathbb{E}_{q_a}[\ln \phi_a] + \mathbb{H}_{\text{BP}}(q). \quad (7)$$

Optimizing  $F_{\text{BP}}(q)$  under the given constraints using the method of Lagrange multipliers yields the update equations

of the BP algorithm. Loosely speaking, the Lagrange multipliers assume the role of messages between variables and factors ( $m_{i \rightarrow a}(x_i)$  and  $m_{a \rightarrow i}(x_i)$ ), each encoding a distribution over the respective variable  $X_i$ . With abuse of notation, writing  $i \in \mathbf{X}_a$  instead of  $X_i \in \mathbf{X}_a$ , the update equations are

$$m_{i \rightarrow a}(x_i) \propto \prod_{b: i \in \mathbf{X}_b, b \neq a} m_{b \rightarrow i}(x_i), \quad (8)$$

$$m_{a \rightarrow i}(x_i) \propto \sum_{x_a \models x_i} \phi_a(x_a) \prod_{j \in \mathbf{X}_a, j \neq i} m_{j \rightarrow a}(x_j). \quad (9)$$

The BP algorithm recomputes the message values according to those equations until convergence (which is not guaranteed). The variable beliefs can then be computed by  $b_i(x_i) = \prod_{a: i \in \mathbf{X}_a} m_{a \rightarrow i}(x_i)$ , and factor beliefs are given by  $b_a(x_a) = \phi_a(x_a) \prod_{i \in \mathbf{X}_a} m_{i \rightarrow a}(x_i)$ .

### 3 VARIATIONAL CBP

Before introducing the LCBP model, we want to interpret CBP in a variational way as a mixture model. For this we reduce the iterative CBP algorithm (Figure 1) to the BP inference on the induced partitioning into cases (the leafs of the tree), and ignore the way in which the partition was obtained. We call this non-iterative interpretation *variational CBP*, and use the term *iterative CBP* when we want to emphasize the recursive conditioning aspect. In variational CBP, we are given a set of partial assignments/conditions  $C$  whose extensions partition the set of all assignments  $\text{Val}(\mathcal{X})$ . The set  $C$  corresponds to the leafs of a tree produced when running iterative CBP. A member  $q$  of class  $\mathcal{Q}_{\text{CBP}}$  is then defined by

$$q(x) = \sum_{c \in C} q_C(c) \prod_a q_a^c(x_a) \prod_i q_i^c(x_i)^{1-d_i}. \quad (10)$$

It can be interpreted as a mixture of BP approximations, where  $q_C(c)$  encodes the mixture weight. The necessary constraints are the BP constraints for each set of beliefs  $q_a^c, q_i^c$ . The weight vector  $q_C : C \rightarrow [0, 1]$  is required to be a proper distribution (non-negative, sum to one). And in addition to the BP constraints, we require  $q_i^c(x_i) = 1$  for  $c \models x_i$  to enforce the conditions within the mixture components. As a result of this constraint, the mixture components have mutually exclusive support. By defining an appropriate energy functional, and solving the variational problem for  $\mathcal{Q}_{\text{CBP}}$ , one finds that a solution can be found by solving the BP variational problem for each mixture component independently.

The computational cost of CBP is about linear in the number of conditions, as each condition implies one run of the BP algorithm. Let us assume that the number of conditioned variables has to attain a certain ratio of the total number of variables for CBP to be able to produce a good approximation. This implies that the number of distinguished

conditions  $|C|$  (and thus inference cost) grows exponentially with problem size when sustaining good approximation quality.

## 4 LCBP

LCBP is designed with the goal that its computational cost scales sub-exponentially in the number of (fully) conditioned variables. This means, we want to approximate variational CBP for an exponentially large set  $C$ , and have the computational cost scaling only polynomially with  $\ln |C|$ . To achieve this we have to overcome two obstacles. The first one is getting rid of the exponential number of parameters  $q_a^c, q_i^c$  present in the variational CBP approximation. Under the assumptions that BP messages do not differ much when far away from a disturbance (Figure 3), we can substitute some  $q_a^{c_1}$  by  $q_a^{c_2}$  in equation 10 given that factor  $\phi_a$  is far enough from all variables where conditions  $c_1$  and  $c_2$  differ. The second problem is representing the weight distribution  $q_C$ . As we will observe in the sequel, this problem will be solved by representing  $q_C$  in factored form, necessitating probabilistic inference over the condition variables.

### 4.1 CONDITIONING SCHEME

To formalize which local functions  $q_a^c, q_i^c$  can be shared between conditions, we introduce a concept termed *conditioning scheme*. We focus on a particular form of conditioning scheme that we call factored, local scheme (FL-scheme). FL-schemes are not powerful enough to capture all aspects of iterative CBP, i.e., they emulate only balanced and statically ordered search trees. But their simple structure allows a formal derivation of the LCBP algorithm, while they are expressive enough to capture the essential improvement LCBP offers over CBP. For a discussion on lifting the restrictions implied by FL-schemes see Section 6.1.

An FL-scheme  $\mathcal{S} : \mathcal{X} \rightarrow 2^{\mathcal{X}}$  assigns a set of conditioning variables (conditioners) to each variable in  $\mathcal{X}$ . The idea is that, locally at a variable  $X_i$ , we have a copy of the BP messages and beliefs for each assignment to the conditioners  $\mathcal{S}(X_i)$  of  $X_i$ . We use the notations  $S_i = \mathcal{S}(X_i)$  and  $S_a = \bigcup_{i \in \mathbf{X}_a} \mathcal{S}(X_i)$  for the set of variable conditioners and factor conditioners respectively. We write  $C = \bigcup_{X_i \in \mathcal{X}} \mathcal{S}(X_i)$  for the set of all conditioners. Given some variable  $X_c$ , we call the set  $\{X_i \mid X_c \in S_i\}$  the area of influence of conditioner  $X_c$  or the set of  $X_c$ 's conditionees. For the lower right example in Figure 4, we have  $C = \{X\}$ . The three variables around  $X$ , and  $X$  itself are the conditionees of  $X$ . They have only  $X$  as their conditioner, and thus are replicated for each possible value of  $X$  (0 and 1). The variable at the left-most corner is not conditioned, and thus has an empty set assigned by the scheme. Note that an FL-scheme only tells how to split variables and the associated variable

beliefs. Factor beliefs are split by assignments to the union of the conditioners of the variables in their scope  $S_a$ . They are thus always split in a more fine-grained way than the adjacent variables.

## 4.2 APPROXIMATING CLASS $\mathcal{Q}_{\text{LCBP}}$

Given an FL-scheme  $\mathcal{S}$  for a problem  $\Phi$ , we define a pseudo distribution  $q$  from class  $\mathcal{Q}_{\text{LCBP}}$ . The parameters are the variable beliefs  $q_i^{c_i}$  and factor beliefs  $q_a^{c_a}$  known from BP, but now each in multiple versions for each local variable condition  $c_i \in \text{Val}(S_i)$  or local factor condition  $c_a \in \text{Val}(S_a)$ . In addition we require a normalized probability measure  $q_C : \text{Val}(C) \rightarrow [0, 1]$  over all possible conditions. Until the end of this section we assume that  $q_C$  is represented as a flat (unstructured) function. Writing  $x[A]$  for restricting the assignment  $x \in \text{Val}(\mathcal{X})$  to the variables in  $A \subseteq \mathcal{X}$ , we define one mixture component  $q^c$  as

$$q^c(x) = \prod_a q_a^{c[S_a]}(x_a) \prod_i \left( q_i^{c[S_i]}(x_i) \right)^{1-d_i}. \quad (11)$$

We define the pseudo distribution for the LCBP model as

$$q(x) = \sum_{c \in \text{Val}(C)} q_C(c) q^c(x). \quad (12)$$

To enforce that  $q$  is close to a probabilistic measure, we formulate a set of constraints on its parameters. Non-negativity constraints are assumed implicitly.

The normalization of the conditioning distribution:

$$\sum_c q_C(c) = 1 \quad (13)$$

The normalization of factor beliefs for all  $\phi_a \in \Phi, c_a \in \text{Val}(S_a)$ :

$$\sum_{x_a} q_a^{c_a}(x_a) = 1 \quad (14)$$

The normalization of variable beliefs for all  $X_i \in \mathcal{X}, c_i \in \text{Val}(S_i)$ :

$$\sum_{x_i} q_i^{c_i}(x_i) = 1 \quad (15)$$

The marginal consistency constraints for all  $\phi_a \in \Phi, X_i \in \mathbf{X}_a, c_i \in \text{Val}(S_i), x_i \in \text{Val}(X_i)$ :

$$\sum_{c_a \models c_i} q_C(c_a | c_i) \sum_{x_a \models x_i} q_a^{c_a}(x_a) = q_i^{c_i}(x_i) \quad (16)$$

We enforce the condition for all  $X_i \in C, c_i \in \text{Val}(S_i)$ :

$$q_i^{c_i}(x_i) = 1 \quad (17)$$

Equation 16 is the LCBP version of the marginal consistency constraints of the BP approximation. It formalizes the way in which beliefs are merged between conditions by taking the expectation with respect to the distribution over conditions  $q_C$ .

## 4.3 FREE ENERGY APPROXIMATION

We are now going to derive a set of fixed-point equations that can be used to implement a message passing algorithm. By partitioning the set of variables into conditioners and the rest  $\bar{C} = \mathcal{X} \setminus C$ , applying the identity  $\mathbb{H}(C, \bar{C}) = \mathbb{H}(C) + \mathbb{H}(\bar{C}|C)$  for the conditioned entropy, and using the Bethe-Peierls approximation (6), we obtain

$$\mathbb{H}_{\text{LCBP}}(q) = \mathbb{H}(q_C) + \mathbb{E}_{q_C}[\mathbb{H}_{\text{BP}}(q^c)]. \quad (18)$$

Under the assumption that the conditioned factor beliefs  $q_a^{c_a}$  are truly the marginals of  $q$  over  $\mathbf{X}_a$  under given condition  $c$ , we can write the energy functional for LCBP as

$$F_{\text{LCBP}}(q) = \sum_a \sum_{c_a} q_C(c_a) \sum_{x_a} q_a^{c_a}(x_a) \ln \phi_a(x_a) + \mathbb{H}_{\text{LCBP}}(q). \quad (19)$$

## 4.4 UPDATE EQUATIONS

Optimizing (19) under the constraints (13) to (17) using the method of Lagrange multipliers lets us derive the message update rules<sup>1</sup>. The update equations work on these additional entities:

1.  $m_{i \rightarrow a}^{c_i}(x_i)$  is a message from variable  $i$  to factor  $a$  under variable condition  $c_i \in \text{Val}(S_i)$ .
2.  $m_{a \rightarrow i}^{c_a}(x_i)$  is a message from factor  $a$  to variable  $i$  under variable condition  $c_i \in \text{Val}(S_i)$ .
3.  $n_{a \rightarrow i}^{c_a}(x_i)$  is a message from factor  $a$  to variable  $i$  under factor condition  $c_a \in \text{Val}(S_a)$ .

We use the artificial factor  $\rho_i^{c_i}(x_i) = \mathbf{1}[c_i[X_i] = x_i]$  to enforce condition (17) on the variable beliefs<sup>2</sup>. The update equations are as following:

$$m_{i \rightarrow a}^{c_i}(x_i) \propto \rho_i^{c_i}(x_i) \prod_{b: i \in \mathbf{X}_b, b \neq a} m_{b \rightarrow i}^{c_b}(x_i) \quad (20)$$

$$m_{a \rightarrow i}^{c_a}(x_i) \propto \sum_{c_a \models c_i} q_C(c_a | c_i) \cdot n_{i \rightarrow a}^{c_a}(x_i) \quad (21)$$

$$n_{a \rightarrow i}^{c_a}(x_a) \propto \sum_{x_a \models x_i} \phi_a(x_a) \prod_{j \in \mathbf{X}_a, j \neq i} m_{j \rightarrow a}^{c_a[S_j]}(x_j) \quad (22)$$

The variable and factor beliefs are computed from the messages via the following formulas:

$$q_a^{c_a}(x_a) \propto \phi_a(x_a) \prod_{i \in \mathbf{X}_a} m_{i \rightarrow a}^{c_i}(x_i) \quad (23)$$

$$q_i^{c_i}(x_i) \propto \rho_i^{c_i}(x_i) \prod_{a: i \in \mathbf{X}_a} m_{a \rightarrow i}^{c_a}(x_i) \quad (24)$$

<sup>1</sup>A more detailed derivation of the update equations is provided in the appendix available in the supplied materials.

<sup>2</sup> $\mathbf{1}[A]$  represents the indicator function that yields 1 when the condition  $A$  is true and 0 otherwise.

And the update equation for the condition distribution is

$$q_C(c) \propto \exp [F_{\text{BP}}(q^c)] \prod_a \prod_{i \in \mathbf{X}_a} \prod_{x_i} \delta_{ai}^{c_a}(x_i), \quad (25)$$

with

$$\delta_{ai}^{c_a}(x_i) = m_{i \rightarrow a}^{c_i}(x_i)^{m_{i \rightarrow a}^{c_i}(x_i)(n_{a \rightarrow i}^{c_a}(x_i) - m_{a \rightarrow i}^{c_i}(x_i))}. \quad (26)$$

The fact that the stated update equations are suited to optimize the formulated variational problem is formalized by the following theorem.

**Theorem 1.** *The interior stationary points of the variational problem specified by maximizing the LCBP functional (19) under the given constraints (13) through (17) are exactly the fixed points of the LCBP update equations (20) through (25).*

The proof is given by the derivation in the appendix available in the extended version of this paper.

The term (26) (and thus the triple product in (25)) vanishes when the messages  $m_{a \rightarrow i}^{c_a}$  agree with the aggregate message  $m_{a \rightarrow i}^{c_i}$ . According to Figure 3 this can happen when the set of conditionees is chosen to be large. Empirically we could not detect a significant difference in inference quality between calculating the  $\delta_{ai}^{c_a}$  terms according to (26) or setting them to 1.

Until now we have treated the distribution over the conditions  $q_C(c)$  as flat. Taking a closer look at equation (25), we notice that the right hand side is a product, with factors coming from the exponentiated BP energy and the  $\delta_{ai}^{c_a}$  terms. These factors all depend on different subsets of variables from  $C$ . Thus the right side of equation (25) describes an undirected graphical model. We call it the *condition problem*, while referring to the original problem as the *primal problem*. When calibrating the message beliefs, it becomes necessary to calculate conditional probabilities  $q_C(c_a|c_i)$  for this problem, and this can be done using any inference algorithm for graphical models. The graphical structure of the condition problem is determined by the overlap between the sets of conditionees for different conditioners, and exact inference in the condition problem can become intractable.

## 5 EMPIRICAL EVALUATION

We conducted two experiments examining the performance of LCBP on randomly generated problem instances. The first experiment is meant to demonstrate that the quality of the LCBP approximation approaches that of variational CBP when increasing the area of influence around conditioned nodes. A second experiment examines how the computational effort of LCBP scales when the problem size increases. In both experiments we condition fully on all conditioners to obtain the variational CBP approximation.

For the experiments, we have implemented two variants of LCBP using the derived update equations. The first variant (LCBP-JT) employs exact inference over the condition problem using the Junction tree method (Shenoy and Shafer, 1990). The second variant (LCBP-BP) uses BP to approximate the marginals of the condition problem. All algorithms are implemented using Round-robin message schedules with no damping. Except for plain BP, the algorithms managed to converge every time. When reporting accuracy we remove all instances where BP did not converge, and thus favor BP in our presentation. We like to remark that when running algorithms from the CBP class, the tolerance for the convergence check has to be set very low. Otherwise the numerical errors may pile up and deteriorate the result even below BP level.

For the first set of experiments, we applied LCBP-JT and CBP to  $6 \times 6$  grid problems with binary variables and random interactions (Figure 5). We selected four fixed variables as conditioners. We varied both the number of conditionees and the interaction strength of the random grids. As expected, the error produced by LCBP-JT approaches the error of CBP both with decreasing interaction strength, and with growing area of influence. We can thus conclude that LCBP-JT acts as an approximation to the CBP result.

The second set of experiments is meant to examine the scaling behavior of LCBP. While the number of parameters of the variational approximation of LCBP grows more slowly than variational CBP, it is conceivable that LCBP takes significantly longer to converge (or even fails to converge at all). To be able to demonstrate that LCBP can yield good quality approximations, we designed a special problem class that we call two-layer grid model (Figure 6). Models with a similar geometry are used in image classification (Kato et al., 1996), sometimes called hierarchical Markov random fields. CBP can achieve good results for this class of problems when conditioning on the nodes of the upper layer, if the variables in the remaining problem (the lower layer) interact only weakly and can thus be approximated well by BP. We applied BP, CBP, LCBP-JT and LCBP-BP to two-layer grid problems with weak interaction on the first and second layer, and strong interactions between layers. We varied the size of the problems to examine the computational effort of the various algorithms. The chosen FL-scheme marks all second layer nodes as conditioners, with all directly connected first layer nodes as respective conditionees.

Figure 7a shows the relative error in the inferred log partition function for varying problem sizes. Notice how all examined algorithms maintain about the same approximation quality once boundary effects are overcome; starting with widths greater than 10. We can observe that all conditioning algorithms improve over the plain BP approximation. CBP performs best, followed closely by LCBP-JT. LCBP-BP produces the worst result among the condition-

ing approaches, though its result is still better than BP by about two orders of magnitude. The difference between the LCBP-JT and the LCBP-BP result was expected, since the condition problem is not acyclic—it contains strong dependencies induced by the explicit interactions between the variables in the second layer. Experiments without interactions in the second layer (not shown) put the LCBP-BP result much closer to LCBP-JT, as paths going through the lower grid induce only weak coupling in the condition problem, while experiments with stronger interactions put LCBP-BP closer to the BP results. For all examined parameter combinations the experiment produces the same qualitative result, i.e., the same order among the examined algorithms. Figure 7b shows the CPU time for the different algorithms. As expected CBP shows an exponential growth with problem size and thus the number of conditioners. In contrast, the effort for the LCBP variants grows approximately linearly with problem size. Note that the LCBP implementations underwent only moderate optimization, as we are interested only in their asymptotic behavior. Thus, one should not draw any conclusions from the concrete slopes of the curves in Figure 7b. Also note that the generated problems have fixed tree-width; both the primal problem and the condition problem. This explains the linear scaling of LCBP-JT.

## 6 DISCUSSION AND FUTURE WORK

The presented LCBP algorithm provides a scalable approximation to the variational interpretation of CBP. By shifting our focus from iterative CBP to variational CBP, we have lost the anytime behavior that iteratively refines the approximation over time in a heuristically guided way. The step from variational CBP to LCBP further removed the possibility of having an unbalanced and dynamically ordered tree to represent the set of examined conditions. This was necessary to achieve the factorization of the condition problem. According to our assessment, LCBP can be extended to resemble iterative CBP more closely, although we expect that this requires substantial further work. The situation is not much different for Generalized Belief Propagation (Yedidia et al., 2005), though. An iterative and heuristically guided construction of region graphs for GBP is as desirable as the adaptive construction of conditioning schemes in the LCBP setting. Approaches to this problem for GBP are still rare, although some work does exist (Welling, 2004; Sibel et al., 2012). In this section we will discuss some steps that point in this direction. Note that some of the discussion is also applicable to GBP.

### 6.1 DESIGNING CONDITIONING SCHEMES

When looking at the iterative CBP algorithm, it is apparent that one of its main strengths is its ability to focus its work on the modes of the target distribution. The trees that can

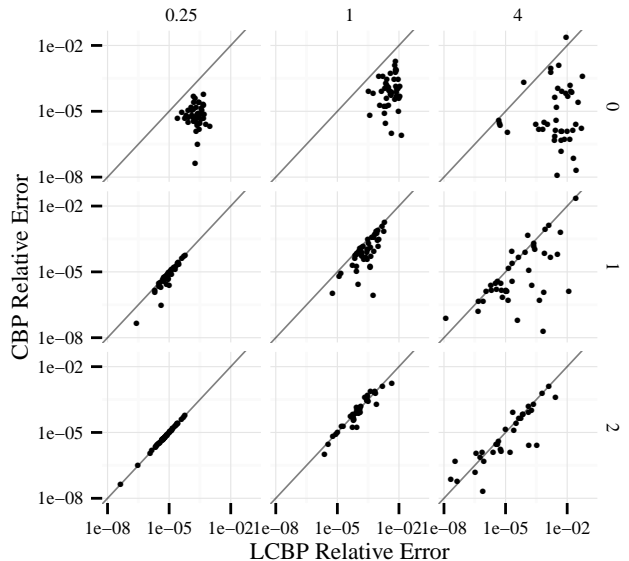


Figure 5: Evaluation results comparing LCBP-JT relative error in log partition function with CBP relative error on randomly generated binary,  $6 \times 6$  grid networks. Four symmetrically placed variables are conditioned. The set of conditionees for LCBP-JT is increased along the rows, including variables with a distance of at most 0, 1 or 2 according to the max norm on the grid coordinates. The columns stand for grids with stronger potentials from left to right (sampled from  $\exp(\mathcal{N}(0, \sigma))$ , with  $\sigma \in \{0.25, 1, 4\}$ ). One can see that both with growing area of influence (going down), and with weaker coupling (right to left!) the result of LCBP-JT approaches that of CBP. If both algorithms disagree, CBP yields a lower error.

be constructed have no constraint on their shape, and they can become very deep and narrow. This is also an advantage of iterative CBP over GBP, where the approximation is improved by using marginals over larger clusters of variables than the factor clusters  $q_a$  used in BP. Given a cluster, GBP can only improve by adding another variable to the cluster, which multiplies the computational burden associated with the cluster by the domain size of the added variable. CBP can circumvent this problem by refining single leaves of its tree; basically making context-specific refinements. LCBP faces the same problem as GBP, because of the limited expressiveness of FL-schemes. With FL-schemes the complexity of LCBP scales exponentially with the number of conditioners a conditionee has—they define the size of the factor scopes for the condition problem. For practical purposes one would aim at using more expressive schemes that allow for context-specific refinements, e.g., conditioning some variable on the conditions  $\{X_1 = 0, X_2 = 0\}, \{X_1 = 0, X_2 = 1\}, \{X_1 = 1\}$ , thus condition only on  $X_2$  for  $X_1 = 0$ . A simple improvement for problems with large variable domains is to branch on elements of arbitrary partitions of assignments to single



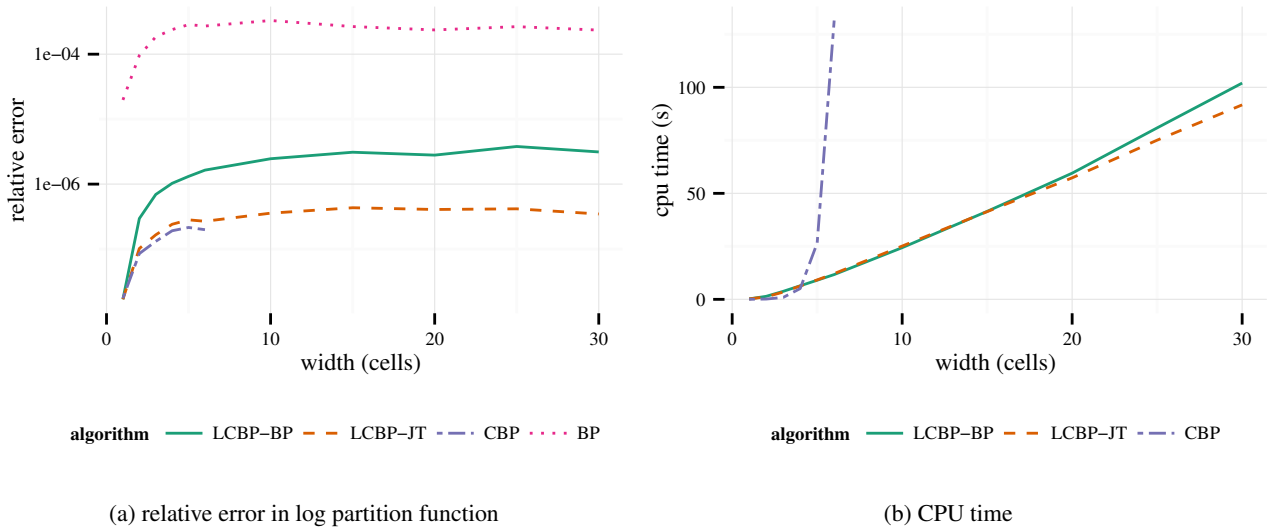


Figure 7: Inference results and used CPU time for two-layer grid problems of varying size (see Figure 6). The x-axis shows the number of columns (width) of the upper grid, where a “cell” is supposed to be a group of nodes on the lower layer connected to a single node on the upper layer. The height of the upper grid is fixed to 2 to obtain problems where exact inference is tractable. All variables have binary domains. Interactions within the lower and the upper layer are weak (factor values drawn from  $\exp\{\mathcal{N}(0, 0.5)\}$ ), while interactions between layers are strong (factor values drawn from  $\exp\{\mathcal{N}(0, 4)\}$ ). CBP was not applied to the larger instances, due to resource constraints. All lines are means over 500 random instances.

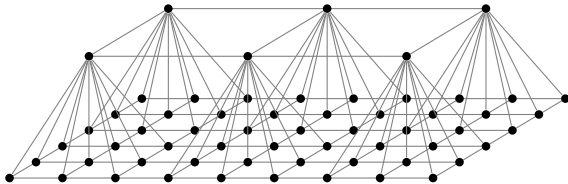


Figure 6: Illustration of the two-layer grid model used for evaluation. Each upper node is connected to  $3 \times 3$  lower nodes. The upper nodes form a two-by- $n$  grid, where  $n$  is varied to obtain problems of different size.

variables, e.g., distinguish between  $X_1 < 3$  and  $X_1 \geq 3$ .

## 6.2 ITERATIVE, HEURISTIC CONSTRUCTION OF SCHEMES

The iterative CBP algorithm has the big advantage of offering an anytime approximation scheme that can be sensitive both to problem structure, and to parameters. This is achieved through the use of different types of heuristics, choosing how to refine the approximation over the course of the computation (Geier et al., 2014b).

Looking at iterative CBP, it appears natural to build the scheme for LCBP incrementally—refining the approximation after running inference and looking at the result. For CBP there exist basically two decision points: Choose the condition/leaf on which to work, then choose the variable to condition on. For LCBP with an FL-scheme choosing

a branch is not possible, as this requires context-specific schemes. If those are available, then LCBP must blur both decision points of CBP into one: Choose which variable to refine under which condition—as the available conditions depend on the chosen variable. This is in contrast to CBP, where all variables are available under every condition (unless they are already conditioned). In addition, for LCBP there exists the new choice of extending the set of conditionees of a conditioner. For this decision we can think of promising candidates for evaluation, like the disagreement among the aggregated messages of the sub-conditions. Using this heuristic would result in splitting variables on the condition until the effect of conditioning has fallen below some threshold (remember Figure 3). Clearly this requirement is too strong, as for tree-structured problems the messages under different conditions can be combined at any moment while still obtaining an exact result.

To find truly informed heuristics, we have to look at the source of the error within the BP approximations. A promising way to construct heuristics for iteratively refining LCBP appears to be exploitation of the loop series expansion (Montanari and Rizzo, 2005). It specifies a correction for the BP functional (Equation 4), that allows to reconstruct the exact value of the partition function. This is done by adding a term for each generalized loop of the graph. Since error contribution is associated with loops, it is not focused on variables, but decentralized. By conditioning on one variable of a loop, while placing the complete loop in the area of influence, the loop can be cor-

rected. In this way the loop series expansion could provide guidance on choosing both conditioners *and* conditionees consistently in an error-oriented manner.

### 6.3 THE CONDITION PROBLEM

One nice aspect about FL-schemes is their property to induce ordinary Markov networks as condition problem. As demonstrated in the evaluation, one can use any algorithm that computes (conditional) marginal probabilities for Markov networks to solve the condition problem. This choice can be influenced by the expected characteristics of the condition problem. If the primal problem contains deterministic dependencies, it is conceivable to “pre-solve” the condition problem. When inference during pre-solving assigns zero probability to some marginal assignments, the corresponding elements of the LCBP calculation can be safely pruned. In addition, elements with very low marginal probability can be pruned on a heuristic basis, incurring a further approximation of the final result.

If a message passing algorithm is chosen for inference within the condition problem, it becomes possible to run it interleaved with the LCBP message updates. This opens the door to using more sophisticated message update schedules, for example Residual Belief Propagation (Elidan, 2006), making it possible to balance the ratio of LCBP updates against inference in the condition problem.

An interesting idea is recursively using LCBP for inference in the condition problem. A perceivable application are hierarchical grid problems with more layers. We expect this construction to scale well, meaning that nesting analysis by cases using LCBP does not incur an exponential growth in model size. It is not clear how to create such a deep hierarchical approximation using other variational techniques, such as GBP.

## 7 RELATED WORK

There exists some prior work on using mixture models for variational inference. Jaakkola and Jordan (1998) use mixtures of mean field approximations to improve inference quality. Beside the weaker approximation of mean field compared to BP and the locality of conditions, the main difference to LCBP is the use of mixture components with overlapping support in contrast to mutually exclusive conditions. The overlapping approach is more powerful in theory, because the mixture components are not restricted in the sense that they are clamped to an intended condition. But in contrast to this, only a weaker approximation to the entropy is used by Jaakkola and Jordan (1998) as the mutually exclusiveness allows for better analytical treatment. Split Variational Inference (Bouchard and Zoeter, 2009) is another application of conditioning and the variational method applied to arbitrary integrals.

The “Gates” model (Minka and Winn, 2008) is also intended as a variational treatment of local mixture components, and arrives at similar update equations for expectation propagation and variational message passing. The LCBP model can be described using Gates with the conditioners being the selector variables, and the conditionees (and incident factors) being placed inside the gate. The LCBP derivation is more explicitly cast as a variational problem by specifying the variational distribution and the constraints, and, more importantly, it allows overlap between gates, which Minka and Winn explicitly forbid. One could say that FL-schemes are more expressive than the (implicit) schemes allowed by Minka and Winn.

## 8 CONCLUSION

We have formulated a variational interpretation of CBP as a mixture of BP approximations. Based on this, we have derived LCBP, which yields inference results that approximate those obtained from CBP. We have shown empirical evidence that supports the claims that LCBP approximates CBP, while scaling much more favorably with problem size.

LCBP allows a non-trivial integration between an arbitrary probabilistic inference algorithm used for solving the condition problem and BP used for inference over the remainder. The automatic construction of good conditioning schemes for LCBP remains an open research question. But we were able to construct schemes for a motivated problem class resembling hierarchical Markov random fields, which are used in image recognition. We are currently working on formulating more expressive classes of conditioning schemes, together with an informed heuristic based on the loop series expansion for BP. We also plan to investigate the relationship between LCBP and GBP more closely.

### Acknowledgements

This work was done within the Transregional Collaborative Research Centre SFB/TRR 62 “Companion-Technology for Cognitive Technical Systems” funded by the German Research Foundation (DFG).

### References

- Bacchus, F., S. Dalmao, and T. Pitassi (2002). Value elimination: Bayesian inference via backtracking search. In *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence*, pp. 20–28.
- Bayardo Jr, R. J. and J. D. Pehoushek (2000). Counting models using connected components. In *Proceedings of the 17th National Conference on Artificial Intelligence*, pp. 157–162.
- Bidyuk, B. and R. Dechter (2007). Cutset sampling for

- Bayesian networks. *Journal of Artificial Intelligence Research* 28, 1–48.
- Bouchard, G. and O. Zoeter (2009). Split variational inference. In *Proceedings of the 26th International Conference on Machine Learning*, pp. 57–64. ACM.
- Darwiche, A. (2001). Recursive conditioning. *Artificial Intelligence* 126(1), 5–41.
- Davis, M., G. Logemann, and D. Loveland (1962). A machine program for theorem-proving. *Communications of the ACM* 5(7), 394–397.
- Dechter, R. (1990). Enhancement schemes for constraint processing: Backjumping, learning, and cutset decomposition. *Artificial Intelligence* 41(3), 273–312.
- Eaton, F. and Z. Ghahramani (2009). Choosing a variable to clamp: Approximate inference using conditioned belief propagation. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics*, Volume 5, pp. 145–152.
- Elidan, G. (2006). Residual belief propagation: Informed scheduling for asynchronous message passing. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*.
- Geier, T., F. Richter, and S. Biundo (2014a). Conditioned belief propagation revisited. In *Proceedings of the 21st European Conference on Artificial Intelligence*, pp. 1011–1012.
- Geier, T., F. Richter, and S. Biundo (2014b). Conditioned belief propagation revisited: Extended version. Technical Report UIB 2014-03, Ulm University.
- Gogate, V. and R. Dechter (2011). SampleSearch: Importance sampling in presence of determinism. *Artificial Intelligence* 175(2), 694–729.
- Huang, J. and A. Darwiche (2005). DPLL with a trace: From SAT to knowledge compilation. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, Volume 5, pp. 156–162.
- Jaakkola, T. S. and M. I. Jordan (1998). Improving the mean field approximation via the use of mixture distributions. In M. Jordan (Ed.), *Learning in Graphical Models*, Volume 89, pp. 163–173. Springer.
- Katebi, H., K. A. Sakallah, and J. P. Marques-Silva (2011). Empirical study of the anatomy of modern SAT solvers. In *Theory and Applications of Satisfiability Testing*, pp. 343–356. Springer.
- Kato, Z., M. Berthod, and J. Zerubia (1996). A hierarchical Markov random field model and multitemperature annealing for parallel image classification. *Graphical models and image processing* 58(1), 18–37.
- Koller, D. and N. Friedman (2009). *Probabilistic Graphical Models: Principles and Techniques*. MIT Press.
- Marques-Silva, J., I. Lynce, and S. Malik (2009). *Handbook of satisfiability*, Chapter CDCL Solvers, pp. 131–150. IOS Press.
- Minka, T. (2005). Divergence measures and message passing. Technical report, Microsoft Research.
- Minka, T. and J. Winn (2008). Gates. In *Advances in Neural Information Processing Systems*, pp. 1073–1080.
- Montanari, A. and T. Rizzo (2005). How to compute loop corrections to the Bethe approximation. *Journal of Statistical Mechanics: Theory and Experiment* 2005(10), 10011.
- Mooij, J. M. and H. J. Kappen (2007). Sufficient conditions for convergence of the sum-product algorithm. *IEEE Transactions on Information Theory* 53(12), 4422–4437.
- Pearl, J. (1986). Fusion, propagation, and structuring in belief networks. *Artificial Intelligence* 29(3), 241–288.
- Roth, D. (1996). On the hardness of approximate reasoning. *Artificial Intelligence* 82(1), 273–302.
- Sang, T., F. Bacchus, P. Beame, H. A. Kautz, and T. Pitassi (2004). Combining component caching and clause learning for effective model counting. In *Proceedings of the 7th International Conference on Theory and Applications of Satisfiability Testing*.
- Shenoy, P. P. and G. Shafer (1990). Axioms for probability and belief-function propagation. In *Proceedings of the 6th Conference on Uncertainty in Artificial Intelligence*, pp. 169–198.
- Sibel, J.-C., S. Reynal, and D. Declercq (2012). A novel region graph construction based on trapping sets for the generalized belief propagation. In *International Conference on Communication Systems (ICCS)*, pp. 305–309. IEEE.
- Silva, J. P. M. and K. A. Sakallah (1996). GRASP—a new search algorithm for satisfiability. In *Proceedings of the International Conference on Computer-Aided Design*, pp. 220–227. IEEE.
- Wainwright, M. J. and M. I. Jordan (2008). Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning* 1(1-2), 1–305.
- Welling, M. (2004). On the choice of regions for generalized belief propagation. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pp. 585–592.
- Yedidia, J. S., W. T. Freeman, and Y. Weiss (2005). Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory* 51(7), 2282–2312.

---

# Discriminative Switching Linear Dynamical Systems applied to Physiological Condition Monitoring

---

**Konstantinos Georgatzis**  
School of Informatics  
University of Edinburgh  
k.georgatzis@sms.ed.ac.uk

**Christopher K. I. Williams**  
School of Informatics  
University of Edinburgh  
ckiw@inf.ed.ac.uk

## Abstract

We present a Discriminative Switching Linear Dynamical System (DSLDS) applied to patient monitoring in Intensive Care Units (ICUs). Our approach is based on identifying the state-of-health of a patient given their observed vital signs using a discriminative classifier, and then inferring their underlying physiological values conditioned on this status. The work builds on the Factorial Switching Linear Dynamical System (FSLDS) (Quinn *et al.*, 2009) which has been previously used in a similar setting. The FSLDS is a generative model, whereas the DSLDS is a discriminative model. We demonstrate on two real-world datasets that the DSLDS is able to outperform the FSLDS in most cases of interest, and that an  $\alpha$ -mixture of the two models achieves higher performance than either of the two models separately.

Condition monitoring of patients in intensive care units (ICUs) based on vital signs (e.g. heart rate, blood pressure) is of critical importance, as they can be subject to a number of serious physiological events such as bradycardia and hypotension. However, a variety of artifactual processes can “contaminate” the data, e.g. the taking of blood samples, performing suction, recalibrating sensors, etc. These artifactual processes complicate the task of identifying the important physiological events and are the main source of false alarms in ICUs. Moreover, it is of interest to maintain beliefs about the true physiological values of a patient when these cannot be directly observed due to artifact. For example, it would be desirable to display the patient’s estimated blood pressure, when the corresponding measuring device has been disconnected or is otherwise displaying artifactual values (as is the case during a blood sample event). Of course, this estimate should be clearly distinguishable from the raw data (e.g. by using a different display colour).

One approach to this problem is to build a latent variable

model, using a number of discrete latent variables to model the physiological and artifactual events through time, and a linear dynamical system (LDS) conditional on these discrete variables to model the associated dynamics in the vital signs observations. This is the factorial switching LDS (or FSLDS) of Quinn *et al.* (2009). However, we have noticed that in building such systems it is necessary to construct quite detailed models of the artifactual events in order to capture them properly. This can be non-trivial since some of these events can be highly variable, which is hard to capture with a generative model. Despite this high variability, the vital signs can still contain informative features which could act as input to a discriminative model. Thus, if it is possible to build such a model that can fairly easily distinguish between the various events, then it would seem simpler and easier to make the discrete-state inference be discriminative, and use FSLDS-style inference for the continuous latent variables conditional on the inferred discrete state. We call this a *discriminative switching linear dynamical system* (DSLDS). In this paper we compare the FSLDS and DSLDS models on two ICU condition monitoring datasets. The results show that using the DSLDS gives increased performance in most cases of interest, and that an  $\alpha$ -mixture of the two methods was able to achieve a higher performance than either of the two models separately.

To summarise, our goal is to build a model with increased performance for the following tasks:

- Identifying artifactual processes (e.g. blood samples), which will reduce the high false alarm rate in ICUs and facilitate the task of identifying physiological processes.
- Identifying physiological processes which can be of critical importance (e.g. bradycardias).
- Providing an estimate of a patient’s true physiological values when these are obscured by artifact.

The structure of the remainder of the paper is as follows: in Section 1 we give a description of our proposed model and

compare its graphical structure and inference methods to those of the FSLDS, and briefly describe related work. In Section 2 we describe our experiments and provide results for the comparison between the DSLDS and the FSLDS. Finally, in Section 3 we conclude with general remarks about our proposed model and suggestions for future work.

## 1 MODEL DESCRIPTION

The graphical model of the FSLDS is depicted in Figure 1 (top). It operates on three different sets of variables: The observed variables,  $\mathbf{y}_t \in \mathbb{R}^{d_y}$  represent the patient’s vital signs obtained from the monitoring devices at time  $t$ , which act as the input to our model. The continuous latent variables,  $\mathbf{x}_t \in \mathbb{R}^{d_x}$ , track the evolution of the dynamics of a patient’s underlying physiology. The discrete variable,  $s_t$ , represents the switch setting or regime which the patient is currently in (e.g. stable, a blood sample is being taken etc.). The switch variable can be factorised according to the cross-product of  $M$  factors, so that  $s_t = f_t^1 \otimes f_t^2 \otimes \dots \otimes f_t^M$ . Each factor variable,  $f_t^m$ , is usually a binary vector indicating the presence or absence of a factor, but in general it can take on  $L^{(m)}$  different values and  $K = \prod_{m=1}^M L^{(m)}$  is the total number of possible configurations of the switch variable,  $s_t$ . Also,  $s_t$  depends explicitly on the previous time step, so that  $p(s_t|s_{t-1}) = \prod_{m=1}^M p(f_t^m|f_{t-1}^m)$ . Conditioned on a particular regime, the FSLDS is equivalent to an LDS. The FSLDS can be seen then as a collection of LDS’s, where each LDS models the dynamics of a patient’s underlying physiology under a particular regime, and can also be used to generate a patient’s observed vital signs. An LDS provides a generative framework for modelling our belief over the state space, given observations.

We can alternatively adopt a discriminative view. We start by modelling  $p(s_t|\mathbf{y}_{t-l:t+r})$  with a discriminative classifier, where (features of) observations from the previous  $l$  and future  $r$  time steps affect the belief of the model about  $s_t$ . The inclusion of  $r$  frames of future context is analogous to fixed-lag smoothing in an FSLDS (see e.g. Särkkä, 2013, sec. 10.5). We note that inclusion of future observations in the conditioning set means that the DSLDS will operate with a delay of  $r$  seconds, since an output of the model at time  $t$  can be produced only after time  $t+r$ . Provided that  $r$  is small enough ( $r \leq 10$  in experiments), this delay is negligible compared to the increase in performance. The LDS can also be regarded from a similarly discriminative viewpoint which allows us to model  $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{y}_t)$ . This is similar to the Maximum Entropy Markov Model (MEMM) (McCallum *et al.*, 2000) with the difference that the latent variable is continuous rather than discrete. The main advantage of this discriminative view is that it allows for a rich number of (potentially highly correlated) features to be used without having to explicitly model their distribution or the interactions between them, as is the case in a generative model. A combination of these two discrimina-

tive viewpoints gives rise to the DSLDS graphical model in Figure 1 (bottom). The DSLDS, conditioned on  $s_t$ , can be seen then as a collection of MEMM’s, where each MEMM in the DSLDS plays a role equivalent to that of each LDS in the FSLDS.

The DSLDS can be defined as

$$p(\mathbf{s}, \mathbf{x}|\mathbf{y}) = p(s_1|\mathbf{y}_1)p(\mathbf{x}_1|s_1, \mathbf{y}_1) \times \prod_{t=2}^T p(s_t|\mathbf{y}_{t-l:t+r})p(\mathbf{x}_t|\mathbf{x}_{t-1}, s_t, \mathbf{y}_t) . \quad (1)$$

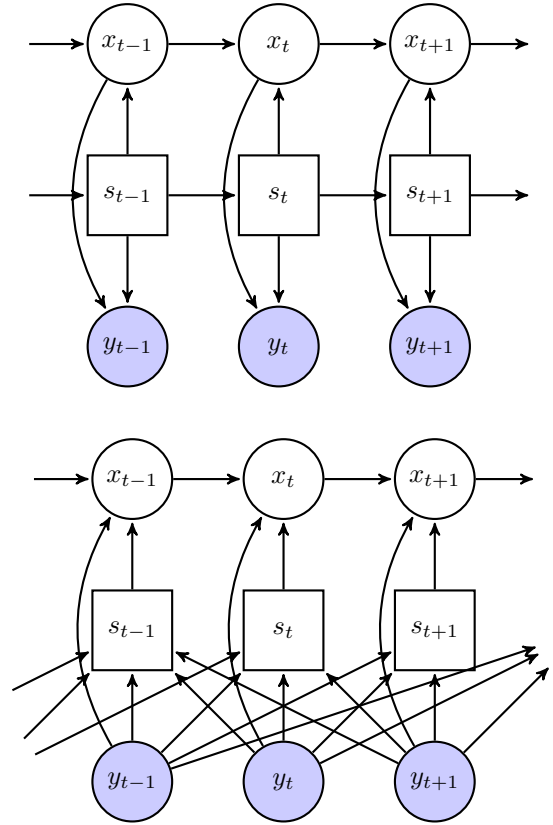


Figure 1: Graphical model of the FSLDS (top) and the DSLDS (bottom). The state-of-health and underlying physiological values of a patient are represented by  $s_t$  and  $\mathbf{x}_t$  respectively. The shaded nodes correspond to the observed physiological values,  $\mathbf{y}_t$ . Note that in the case of the DSLDS the conditional probability  $p(s_t|\mathbf{y}_{t-l:t+r})$  is modelled directly.

The simplest assumption we can make for the DSLDS is that  $p(s_t|\mathbf{y}_{t-l:t+r})$  factorises, so that

$$p(s_t|\mathbf{y}_{t-l:t+r}) = \prod_{m=1}^M p(f_t^{(m)}|\mathbf{y}_{t-l:t+r}) . \quad (2)$$

However, one could use a structured output model to predict the joint distribution of different factors.

## 1.1 Predicting $s_t$

Our belief about the state of health of a patient at time  $t$  is modelled by  $p(s_t | \mathbf{y}_{t-l:t+r})$ , the conditional probability of the switch variable given the observed vital signs. Following the factorisation of the switch variable in eq. 2, we model the conditional probability of each factor being active at time  $t$  given the observations with a probabilistic discriminative binary classifier, so that  $p(f_t^{(i)} = 1 | \mathbf{y}_{t-l:t+r}) = G(\phi(\mathbf{y}_{t-l:t+r}))$ , where  $G(\cdot)$  is a classifier-specific function, and  $\phi(\mathbf{y}_{t-l:t+r})$  is the feature vector that acts as input to our model at each time step as described in Section 2.1. As is evident from Figure 1 (bottom) there is no explicit temporal dependence on the switch variable sequence. However, temporal continuity is implicitly incorporated in our model through the construction of the features.

### 1.1.1 An $\alpha$ -mixture of $s_t$

The DSLDS model can be seen as complementary to the FSLDS, and they can be run in parallel. One way of combining the two outputs is to maintain an  $\alpha$ -mixture over  $s_t$ . If  $p_g(s_t)$  and  $p_d(s_t)$  are the outputs for the switch variable at time  $t$  from FSLDS and the DSLDS respectively, then their  $\alpha$ -mixture is given by:  $p_\alpha(s_t) = c \left( p_g(s_t)^{(1-\alpha)/2} + p_d(s_t)^{(1-\alpha)/2} \right)^{2/(1-\alpha)}$ , where  $c$  is a normalisation constant which ensures that  $p_\alpha(s_t)$  is a probability distribution. The family of  $\alpha$ -mixtures then subsumes various known mixtures of distributions and defines a continuum across them via the  $\alpha$  parameter. For example, for  $\alpha = -1$  we retrieve the mixture of experts (with equally weighted experts) framework, while for  $\alpha \rightarrow 1$ , the formula yields  $p_1(s_t) = c\sqrt{p_g(s_t)p_d(s_t)}$ , rendering it equivalent to a product of experts viewpoint. In general, as  $\alpha$  increases, the  $\alpha$ -mixture assigns more weight to the smaller elements of the mixture (with  $\alpha \rightarrow \infty$  giving  $p_\infty(s_t) = \min\{p_g(s_t), p_d(s_t)\}$ ), while as  $\alpha$  decreases, more weight is assigned to the larger elements (with  $\alpha \rightarrow -\infty$  giving  $p_{-\infty}(s_t) = \max\{p_g(s_t), p_d(s_t)\}$ ). A thorough treatment is given in Amari (2007).

## 1.2 Predicting $\mathbf{x}_t$

The model of the patient’s physiology should capture the underlying temporal dynamics of their observed vital signs under their current health state. The idea is that the current latent continuous state of a patient should be dependent on (a) the latent continuous state at the previous time step, (b) the current state of health and (c) the current observed val-

ues. We model these assumptions as follows

$$p(\mathbf{x}_t | \mathbf{x}_{t-1}, s_t, \mathbf{y}_t) \propto \exp\left\{-\frac{1}{2}(\mathbf{x}_t - \mathbf{A}^{(s_t)}\mathbf{x}_{t-1})^\top (\mathbf{Q}^{(s_t)})^{-1}(\mathbf{x}_t - \mathbf{A}^{(s_t)}\mathbf{x}_{t-1})\right\} \times \exp\left\{-\frac{1}{2}(\mathbf{C}^{(s_t)}\mathbf{x}_t - \mathbf{y}_t)^\top (\mathbf{R}^{(s_t)})^{-1}(\mathbf{C}^{(s_t)}\mathbf{x}_t - \mathbf{y}_t)\right\} . \quad (3)$$

The first term on the RHS of eq. 3 is the **system model** for an LDS and captures the dynamics of a patient’s latent physiology under state  $s_t$ . The second term can be seen as the discriminative counterpart of the **observation model** of an LDS. In our condition monitoring setting, the observed vital signs are considered to be noisy realisations of the true, latent physiology of a patient and thus, the observation model encodes our belief that  $\mathbf{x}_t$  is a noisy version of  $\mathbf{y}_t$ . Under this assumption,  $\mathbf{C}^{s_t}$  consists of 0/1 entries, which are set based on our knowledge of whether the observations  $\mathbf{y}_t$  are artifactual or not under state  $s_t$ . In the FSLDS, the corresponding observation model encodes the belief that the generated  $\mathbf{y}_t$  should be normally distributed around  $\mathbf{x}_t$  with covariance  $\mathbf{R}^{s_t}$ , whereas in our discriminative version, the observation model encodes our belief that  $\mathbf{x}_t$  should be normally distributed around  $\mathbf{y}_t$  with covariance  $\mathbf{R}^{s_t}$ . The idea behind this model is that at each time step we update our belief about  $\mathbf{x}_t$  conditioned on its previous value,  $\mathbf{x}_{t-1}$ , and the current observation,  $\mathbf{y}_t$ , under the current regime  $s_t$ . For example, under an artifactual process, the observed signals do not convey useful information about the underlying physiology of a patient. In that case, we drop the connection between  $\mathbf{y}_t$  and  $\mathbf{x}_t$  (for the artifact-affected channels) which translates into setting the respective entries of  $\mathbf{C}^{s_t}$  to zero. Then, the latent state  $\mathbf{x}_t$  evolves only under the influence of the appropriate system dynamics parameters  $(\mathbf{A}^{(s_t)}, \mathbf{Q}^{(s_t)})$ . Conversely, operation under a non-artifactual regime incorporates the information from the observed signals, effectively transforming the inferential process for  $\mathbf{x}_t$  into a product of two “experts”, one propagating probabilities from  $\mathbf{x}_{t-1}$  and one from the current observations.

We note that the step of conditioning on the current regime  $s_t$  in order to predict  $\mathbf{x}_t$  is required for our task, as we do not have training data for the  $\mathbf{x}$ -state. Otherwise, one could imagine building a simpler model such as a conditional random field (Lafferty *et al.*, 2001), to predict the  $\mathbf{x}$ -state directly from the observations. However, in our case, where only labels about the patient’s regime are available, this is not possible.

## 1.3 Learning

We first describe learning in the general SLDS setting. The parameters that need to be learned are:  $\{\mathbf{A}^s, \mathbf{Q}^s, \mathbf{C}^s, \mathbf{R}^s\}$ . Given training data for each switch setting, these can be learned independently as LDS parameters for each configuration of  $s$ . Following Quinn *et al.* (2009) we use an

independent ARIMA model with added observation noise for each channel. Casting such a model into state space form is a standard procedure as described in Brockwell and Davis (2009, sec. 12.1), and amounts into reformulating the parameters of the ARIMA model into the parameters of a state-space model. Once the model is in state space form,  $\mathbf{A}^s$ ,  $\mathbf{Q}^s$ ,  $\mathbf{C}^s$ ,  $\mathbf{R}^s$  can be fit according to the maximum likelihood criterion by using numerical optimisation methods (like Newton-Raphson, Gauss-Newton), as presented in Shumway and Stoffer (2000, sec. 2.6) or expectation maximisation (EM) as presented in Ghahramani and Hinton (1996). We note that the vector ARMA (VARMA) representation is used, where for example a one-dimensional AR( $p$ ) process can be encoded as a  $p + 1$ -dimensional VAR(1) process by maintaining a latent state representation of the form  $\mathbf{x}_t = [x_t \ x_{t-1} \ \dots \ x_{t-p}]$ .

In the DSLDS, the same set of parameters needs to be learned. As mentioned in Section 1.2, the assumptions for the DSLDS observation model constrain  $\mathbf{C}^s$  to be a binary matrix, whose values are set so as to pick the most recent value  $\mathbf{x}_t$  under the VARMA representation. For example, assuming that we are modelling one channel, under a physiological regime, as an AR(2) process, then  $\mathbf{C}^s = [1 \ 0 \ 0]$ . Under this constrained form of  $\mathbf{C}^s$  we obtain the remaining parameters,  $\mathbf{A}^s$ ,  $\mathbf{Q}^s$  and  $\mathbf{R}^s$ , using the same learning process as the one already described for the case of a general SLDS.

The task of determining the order of the respective ARIMA models is less straightforward. We have followed a practical approach as suggested in Diggle (1990, sec. 6.2). The autocorrelation and partial autocorrelation function (ACF and PACF respectively) of the stationary data (if a time series is not stationary, we make it stationary by successive differencing) were examined to provide an initial estimate of the appropriate model order. A clear cut-off at lag  $q$  in the ACF plot is suggestive of an MA( $q$ ) process, while a clear cut-off at lag  $p$  in the PACF plot is suggestive of an AR( $p$ ) process. Clear cut-offs are rare in a real world application, in which case we looked for less clear tail-offs in the PACF and ACF plots. After establishing a small number of potential model orders suggested by these tail-offs, further exploration of the model order around these initial estimates was carried out by calculating the Akaike Information Criterion (AIC) score (Akaike, 1972) for each of these potential model orders, and finally the one with the smallest AIC value was chosen.

#### 1.4 Inference

In this paper we are concerned with the task of computing the distribution  $p(s_t, \mathbf{x}_t | \mathbf{y}_{1:t+r})$ . According to our proposed model,  $p(s_t | \mathbf{y}_{t-l:t+r})$  can be inferred at each time step via a classifier as described in Section 1.1. However, exact inference for  $\mathbf{x}_t$  is still intractable. The same lim-

itation as in the case of a standard SLDS applies (Lerner and Parr, 2001): In order to maintain an exact belief over the posterior distribution of  $\mathbf{x}_t$  we need to keep track of all the potential combinations of switch variable settings that could have lead us from  $\mathbf{x}_{t-1}$  to  $\mathbf{x}_t$ , making inference scale exponentially with time. An approximation of this distribution can be maintained via the Gaussian Sum algorithm<sup>1</sup> (Alspach and Sorenson, 1972). The idea is that at each time step  $t$  we maintain an approximation of  $p(\mathbf{x}_t | s_t, \mathbf{y}_{1:t+r})$  as a mixture of  $J$  Gaussians. Moving one time step forward will result in the posterior  $p(\mathbf{x}_{t+1} | s_{t+1}, \mathbf{y}_{1:t+r+1})$  having  $KJ$  components, which are again collapsed to  $J$  components. In our experiments we use  $J = 1$ , which translates into matching moments (up to second order) of the distribution for each setting of  $s_t$ , as shown in Murphy (1998). Therefore inference in the DSLDS can be seen as a two-step process, where  $p(s_t | \mathbf{y}_{t-l:t+r})$  is inferred by our discriminative classifier, and  $p(\mathbf{x}_t | s_t, \mathbf{y}_{1:t+r})$  is inferred according to the Gaussian Sum algorithm.

#### 1.5 Related work

In terms of methodology, our proposed model bears some similarities to the one used by Lu *et al.* (2009). However, their model was used to model spatial relationships and they were only concerned with a binary discrete latent space. In our case, we are concerned with modelling temporal structure and we have a richer and more complex discrete latent space. More importantly, in their work the distribution maintained over the continuous latent space is a single multivariate Gaussian, whereas in our model, as described in the previous section, the belief over the continuous latent space is modelled as a mixture of  $KJ$  Gaussians. This allows us to keep track of multiple modes about the belief over a patient’s underlying physiology, since this is potentially affected by multiple factors.

In terms of application, our work is mostly similar to the one presented in Quinn *et al.* (2009). The same task of inferring artifactual and physiological processes was considered there. However a generative approach was taken there via the use of an FSLDS. In Lehman *et al.* (2014), a switching vector autoregressive model was used on minute-by-minute heart rate and blood pressure vital signs to provide inputs for a logistic regression classifier with the goal of patient outcome prediction. Also, Nemati *et al.* (2013) propose training a SLDS in a discriminative manner so as to optimize prediction of the  $s$  sequence given the observations, and apply this to identifying four postural categories under a controlled protocol. Stanculescu *et al.* (2014) use a hierarchical structure in the discrete space of an SLDS motivated by expert knowledge on modelling sepsis. In our work, we use a discriminative SLDS, capable of mod-

<sup>1</sup>The Gaussian Sum algorithm is also known as the Generalised Pseudo Bayesian (GPB) algorithm as mentioned in Murphy (1998).

elling both discrete and continuous latent states in a unified framework, applied to two challenging real-world datasets. It yields superior results for state-of-health identification, and maintains at the same time beliefs about a patient’s underlying physiology.

## 2 EXPERIMENTS

In this section we describe experiments on two challenging datasets comprising of patients admitted to ICUs in two different hospitals, namely a neonatal ICU and an adult ICU. We emphasise that it is highly non-trivial to obtain annotations for medical datasets as it requires the very scarce resource of experienced clinicians. Indeed, for the adult ICU, the annotated data are the product of a one-year collaboration with that ICU. Physionet (Goldberger *et al.*, 2000), a freely available medical dataset, is not suitable for our task since the only available time-series annotations are a limited set of life threatening/terminal events, for which identification would not be of practical use in the ICU.

For both datasets, we evaluate the performance of the DSLDS compared to the FSLDS. We also report the performance of an  $\alpha$ -mixture of the two models. Note that the FSLDS has been shown in Quinn *et al.* (2009) to achieve superior results compared to more basic models such as a factorial hidden Markov model (FHMM) for the task of condition monitoring in ICUs. We first provide a short description of the various features that were used as input to the state-of-health model as described in Section 1.1, followed by an outline of the main characteristics of the two datasets. We conclude this section by providing results on two tasks: a) inferring a patient’s state of health and b) inferring a patient’s underlying physiology in the presence of artifact corruption.

### 2.1 Features & Classifiers

As described in Section 1.1, the estimate of  $s_t$  is the output of a discriminative classifier. For both datasets, we found that using a random forest (Breiman, 2001) as our classification method yields the best performance. Suggestions for judicious selection of various tree-construction parameters can be found in Hastie *et al.* (2009, Ch. 15). The Gini index was used as the criterion for splitting nodes for each tree in the random forest. The output of the random forest for a new test point is an average of the predictions produced by each tree, where the prediction of each tree is the proportion of the observations that belong to the positive class in the leaf node in which the test point belongs to. Apart from their high performance, another appealing property of random forests is that they can handle missing observations via the construction of surrogate variables and splits within each decision tree as explained in Hastie *et al.* (2009, sec. 9.2.4).

We use a variety of features to capture interesting temporal structure between successive observations. At each time step, a sliding window of length  $l + r + 1$  is computed. For some features we also divide the window into further sub-windows and extract additional features from them. More precisely, the full set of features that are being used are: (i) the observed, raw values of the previous  $l$  and future  $r$  time steps ( $\mathbf{y}_{t-l:t+r}$ ); (ii) the slopes (calculated by least squares fitting) of segments of that sliding window that are obtained by dividing it in segments of length  $(l + r + 1)/k$ ; (iii) an exponentially weighted moving average of this window of raw values (with a kernel of width smaller than  $l + r + 1$ ); (iv) the minimum, median and maximum of the same segments; (v) the first order differences of the original window; and (vi) differences of the raw values between different channels.

### 2.2 Neonatal ICU

The first dataset is the one used in Quinn *et al.* (2009)<sup>2</sup>. It comprises 24-hour periods from fifteen neonates admitted to the ICU of the Edinburgh Royal Infirmary, with events of interest annotated by two clinical experts. These annotations include: i) blood sample events (BS), ii) periods during which an incubator is open (IO), iii) core temperature probe disconnections (TD), iv) bradycardias (BR), and v) periods that are clearly not stable but no further identification was made by the clinicians (X). These last cases can be collectively considered as a “none-of-the-above” factor, which is referred to as the X-factor by Quinn *et al.* (2009). More details about the events of interest can be found in the aforementioned work. We used the same parameters for the underlying physiology model as the ones used there.

### 2.3 Adult ICU

The second dataset comprises data collected from nine adults admitted to the neuro ICU of the Southern General Hospital in Glasgow. An average of 33-hour periods were collected from each of these patients, consisting of measurements recorded on a second-by-second basis for four different channels: heart rate (HR), systolic and diastolic blood pressure (BP<sub>sys</sub>, BP<sub>dia</sub>), and systolic intracranial pressure (ICP<sub>sys</sub>). These data were then annotated by a clinical expert. We give a brief description of the learning process for stability periods and modelled factors, which include blood samples, damped traces (DT), suction events (SC), and the X-factor.

**Stable periods** correspond to time periods when no annotation occurred from the experts, suggesting that the patient is in a stable condition. In Williams and Stanculescu (2011) it was found that in a similar setting a 15 minute period of stability provides an adequate amount of training data. We

<sup>2</sup>The dataset has been anonymised and is available at: [www.cit.mak.ac.uk/staff/jquinn/software.html](http://www.cit.mak.ac.uk/staff/jquinn/software.html)



use the same time interval for our experiments. We found that ARIMA(2,1,0) models were adequate for all channels.

An example of a **blood sample** is shown in Figure 4 (bottom). Changes in BPsyst and BPDia can be modelled as a four-stage process: i) the blood is diverted to a syringe for blood sampling, which causes an artifactual ramp in the observed measurements. This is similar to the blood sample model described in Quinn *et al.* (2009) and we follow the same approach here. ii) A recalibration stage follows, causing measurements to drop to zero which can be modelled similarly to a dropout event as in Quinn *et al.* (2009). iii) BP measurements continue as a stable period for a brief period. iv) The blood sample is concluded with a flushing event for hygiene purposes which causes a sharp increase in measurements. This stage is modelled as an AR(3) process for both the BPsyst and BPDia channels. A total number of 64 blood sample events have been annotated, with an average duration of 1.6 minutes.

During a **suction event**, a flexible catheter is inserted into the airway of the patient to remove secretions that have accumulated over time in their pulmonary system. This event is observed as a significant increase in the values of all observed channels. An AR(2) process models the HR channel, while AR(3) processes were used to model the remaining channels. A total number of 53 suction events have been annotated, with an average duration of 4.3 minutes.

A **damped trace**, an example of which is shown in Figure 4 (top), is usually observed due to blood residues being accumulated in the line used for measuring the blood pressure channel, which leads both BPsyst and BPDia to converge to a similar mean value while at the same time the measurements exhibit high variability. Both channels were modelled with AR(3) processes. A total number of 32 damped trace events have been annotated, with an average duration of 14 minutes.

Except for the aforementioned factors which we explicitly model, there are a multitude of other factors present in our training data, corresponding to either known but not yet modelled factors (such as hygiene events, tachycardias etc.) or to unknown factors (clear abnormalities which however have not been identified by the clinicians). We collectively treat those events as unknown and model them according to the X-factor model proposed in Quinn *et al.* (2009). A total number of 278 X-factor events have been annotated, with an average duration of 7.5 minutes. Channels which are unaffected by an artifactual process (as shown in Table 1) are modelled as in the stable case. In every case the parameters of the x-state models were further optimised by EM.

Table 1: Channels affected by different processes for the adult ICU are marked by ●.

|              | HR | BPsyst | BPDia | ICPsyst |
|--------------|----|--------|-------|---------|
| Blood sample |    | ●      | ●     |         |
| Damped trace |    | ●      | ●     |         |
| Suction      | ●  | ●      | ●     | ●       |
| X-factor     | ●  | ●      | ●     | ●       |

Table 2: Comparison of DSLDS, FSLDS and  $\alpha$ -mixture performance for the Neonatal ICU dataset. Optimal value of the  $\alpha$  parameter is shown inside parenthesis.

| AUC                                | BS   | IO   | TD   | BR   | X    |
|------------------------------------|------|------|------|------|------|
| DSLDS                              | 0.98 | 0.83 | 0.90 | 0.94 | 0.57 |
| FSLDS                              | 0.92 | 0.87 | 0.88 | 0.85 | 0.66 |
| $\alpha$ -mixture <sup>(0.5)</sup> | 0.98 | 0.89 | 0.93 | 0.92 | 0.67 |

## 2.4 Results

For both datasets we compare the performance of the DSLDS and the FSLDS for the task of inferring a patient’s state of health. Having obtained estimates  $s_t$  for each factor and each time step, we proceed to calculate Receiver Operating Characteristic (ROC) curves for the classification of each factor. We also measure the performance of the models by reporting the Area under each ROC curve (AUC). Plots of the ROC curves, comparing the DSLDS, FSLDS, and an  $\alpha$ -mixture of the two models, are shown in Figures 2 and 3.

In the case of the DSLDS, the features described in Section 2.1 involve a number of hyperparameters that need to be chosen. Fitting them with a standard cross-validation (CV) scheme when data are not abundant poses a non-negligible risk of overfitting. As is shown in Varma and Simon (2006), using CV to evaluate performance of a model when the model’s hyperparameters have been themselves tuned using CV can lead to an optimistic bias of the estimate of the true performance. In that same work, a nested CV approach is shown to yield an almost unbiased estimate of the true performance, which we also follow in our experiments. In the outer loop the data are partitioned into  $P$  disjoint test sets. After choosing one of these partitions, the rest of the data are used in the inner loop in a standard CV setup to select the hyperparameters. The hyperparameters which yielded the highest performance (average cross-validated AUC across factors in our case) in the inner loop are then used to estimate the performance of the model on the partition (test set) in the outer loop. This process is repeated  $P$  times, once for each partition in the outer loop. For both datasets, we use leave-one-patient-out CV for the

inner loop and 3-fold CV for the outer loop. In the inner loop, we perform a grid search over hyperparameters in the following sets: a) number of trees of random forest classifiers in  $\{10, 25, 50, 100, 200\}$ ; b)  $l$  in  $\{4, 9, 14, 19, 29, 49\}$ ; c)  $r$  in  $\{0, 5, 10\}$ . The sub-segments lengths (for slope features) were always set to  $\max\{5, (l + r + 1)/5\}$  and the kernel widths (for moving average features) were always set to  $\max\{5, (l + r + 1)/5\}$ .

In the case of the FSLDS, it is not necessary to follow the same procedure. Using the AIC score, as shown in Section 1.3, for choosing the orders of the ARIMA processes (which constitute the model’s hyperparameters) avoids potential overfitting by penalising the model’s likelihood as the parameters grow. We therefore use 3-fold CV to evaluate the FSLDS’s performance.

To evaluate the  $\alpha$ -mixture model, we have chosen the optimal  $\alpha$  value as the one that maximises the average AUC across factors, via 3-fold CV. This also allowed us to explore the behaviour of the model as a function of  $\alpha$  for both datasets.

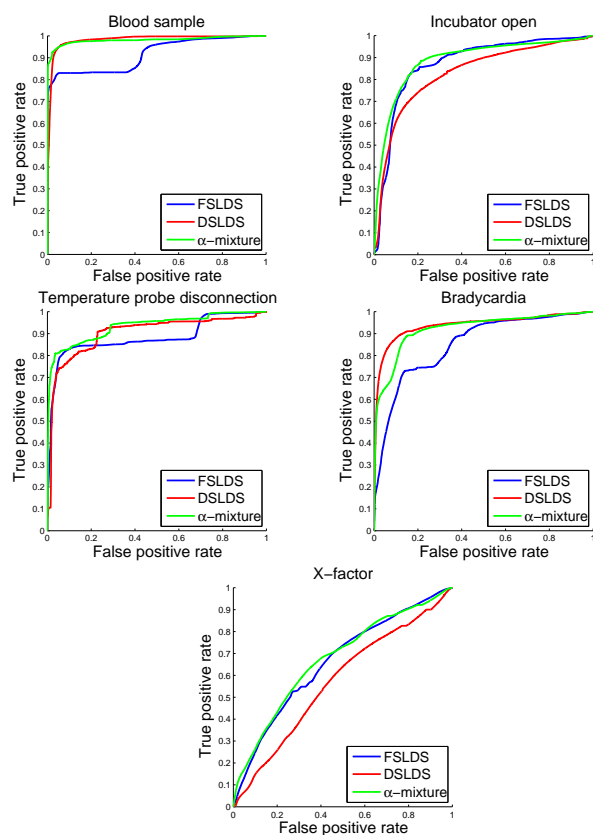


Figure 2: ROC curves per modelled factor in the case of the neonatal ICU.

## 2.4.1 Neonatal ICU

In the case of the neonatal ICU we compare the two models on the full set of annotated factors reported in Quinn *et al.* (2009). The results are shown in Table 2<sup>3</sup>. The DSLDS outperforms the FSLDS in three out of the four clinically identified factors. The difference in favour of the DSLDS is clear for bradycardias and blood samples, but less pronounced for core temperature disconnections. The FSLDS achieves slightly higher performance in the case of the incubator open factor, and clearly outperforms the DSLDS in the case of the X-factor. The FSLDS models the presence of outliers by the inclusion of an extra factor, which is essentially governed by the same parameters as stability with the only difference being that the system noise covariance is an inflated version of the respective covariance of the stability dynamics (for more details, see Quinn *et al.*, 2009). Such an approach has the potential to address the issue of outlier detection in a more general and thus more satisfactory way. In the case of the DSLDS, our approach is to collectively treat all abnormal events, other than the ones attributed to known factors, as an “X-class” and build a binary classifier to distinguish that class. As the training datapoints for this class are highly inhomogeneous in terms of shared discriminative features, and test points belonging to the X-class may not exhibit a high degree of similarity to the training set, it is not surprising that the DSLDS may perform rather poorly for the X-factor. However, by considering an  $\alpha$ -mixture of the two models, we can combine the discriminative power of the DSLDS for known factors with the increased performance of the FSLDS for the X-factor, thus achieving a higher performance (bottom line of Table 2) compared to considering the two models separately. The behaviour of the  $\alpha$ -mixture model as a function of  $\alpha$  is shown in Figure 5 (top). The optimal  $\alpha$ -mixture ( $\alpha = 0.5$ ) yields the best average AUC across factors (in fact,  $\alpha = 0.5$  yields optimal performance for each factor separately except bradycardia, where it is almost optimal) compared to all other considered  $\alpha$  values and also outperforms the DSLDS and the FSLDS in all cases except for the bradycardia factor, where the DSLDS performs slightly better.

## 2.4.2 Adult ICU

In the case of the adult ICU, inferences for two example events are shown in Figure 4. In the top, a damped trace event is shown, which lasts for almost one hour before being resolved by a flushing event (spiking of both channels). The DSLDS accurately identifies the damped trace event,

<sup>3</sup>The FSLDS results were obtained using code provided by Quinn *et al.* (2009) with the same parameters as the ones mentioned there. The results are very close with the exception of the core temperature disconnection factor (for which the reported AUC in Quinn *et al.* (2009) was 0.79, while we obtained a value of 0.88), and the blood sample factor (for which the reported AUC in Quinn *et al.* (2009) was 0.96, while we obtained a value of 0.92).

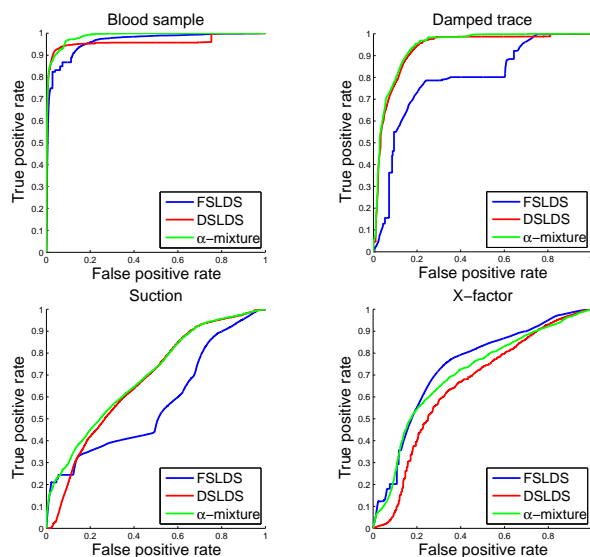


Figure 3: ROC curves per modelled factor in the case of the adult ICU.

Table 3: Comparison of DSLDS, FSLDS and  $\alpha$ -mixture performance for the Adult ICU dataset. Optimal value of the  $\alpha$  parameter is shown inside parenthesis.

| AUC                              | BS   | DT   | SC   | X    |
|----------------------------------|------|------|------|------|
| DSLDS                            | 0.96 | 0.93 | 0.67 | 0.65 |
| FSLDS                            | 0.95 | 0.79 | 0.57 | 0.74 |
| $\alpha$ -mixture <sup>(0)</sup> | 0.99 | 0.94 | 0.70 | 0.71 |

while the FSLDS fails totally to detect it, but hypothesises several incorrect blood sample events. In the bottom panel a blood sample event is shown, where the multiple stages are clearly visible. The event starts with two artifactual ramps, followed by a flushing, a zeroing, and finally with another flushing. This is slightly different than the description we have already given, but slight deviations from the standard protocol due to human error is to be expected. In this case, both models manage to capture the event in a generally satisfactory manner. Summary results are reported in Table 3. The DSLDS outperforms the FSLDS on all of the known factors. The damped trace and suction events particularly are characterised by high variability which is hard to capture with a generative process. However, simple discriminative features are able to capture them with higher accuracy. As was expected, the FSLDS achieves a higher AUC for the X-factor. Again, the optimal  $\alpha$ -mixture ( $\alpha = 0$ ) outperforms the DSLDS and the FSLDS in all cases except for the X-factor, where the FSLDS achieves a slightly higher AUC. Contrary to the neonatal ICU dataset, as shown in Figure 5 (bottom) there are alternative  $\alpha$  values

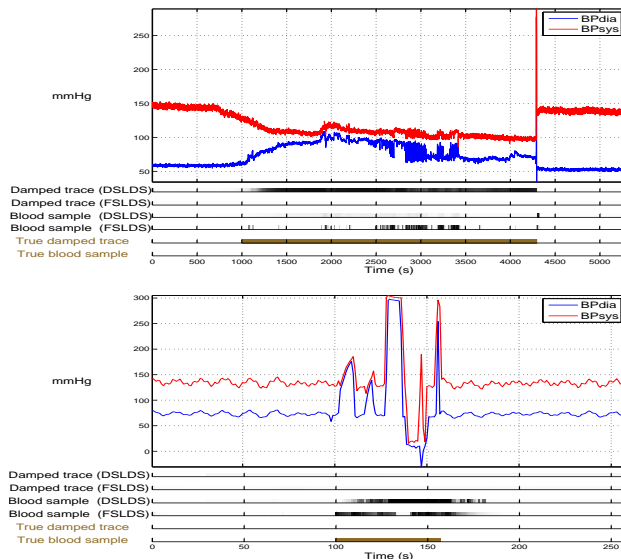


Figure 4: Example of DSLDS and FSLDS inferences for a damped trace event (top) and a blood sample event (bottom).

which can yield higher AUC across different factors. For example, an X-factor AUC value of 0.76 can be obtained by setting  $\alpha = 5$ . However, apart from the superior (on average) performance of the  $\alpha$ -mixture, another appealing property is that  $\alpha$  could be treated as a user-tunable parameter. In a practical setting, the model could be preset with the optimal  $\alpha$  value, but a clinician could decide, for example, to make the model focus on maximising its predictive performance on the X-factor (or some important physiological factor like bradycardia) to the potential detriment of other factors. Then the model could adjust its  $\alpha$  parameter in real-time based on training data results to maximise its performance on the desired factor.

### 2.4.3 Inference for x-state

Finally, Figure 6 shows the inferred distribution of underlying physiology during a blood sample taken from a neonate for both models. In both cases, estimates are propagated with increased uncertainty under the correctly inferred artifactual event. Note a small difference at the start of the event: The DSLDS partially identifies the event causing an increase in uncertainty, while the FSLDS (incorrectly) identifies this part as stable and thus its x-state update exhibits lower uncertainty. Maintaining an estimate of the underlying vital signs in the presence of artifacts can then be used for data imputation. Another use, which has been deemed important by our clinical experts, is that such an estimate can help doctors maintain an approximate view of a patient’s underlying physiology during artifactual events that would otherwise completely obscure a patient’s vital signs. This can be crucial during treatment of a patient un-

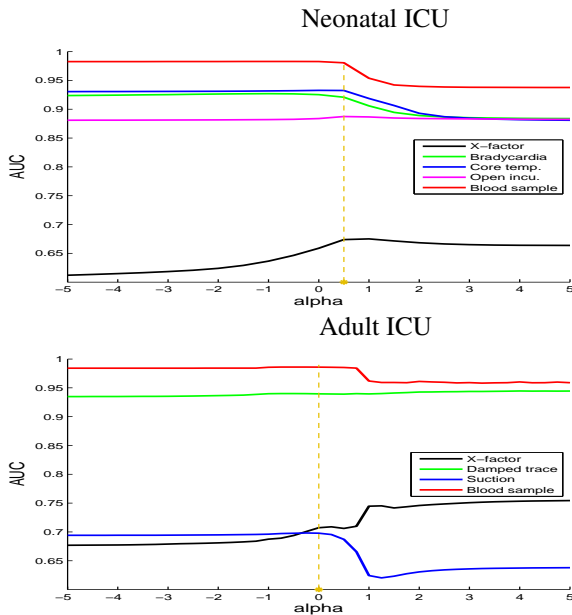


Figure 5: Performance of the  $\alpha$ -mixture models as a function of  $\alpha$  ( $step = 0.25$ ) for the Adult ICU (top) and the neonatal ICU dataset (bottom). The asterisk marks the optimal value for  $\alpha$ .

der critical conditions, such as the ones found in an ICU.

### 3 DISCUSSION

We have presented a discriminative approach for the very important application of patient monitoring in ICUs. We show that our new approach is able to outperform the previous generative approach used for the same task in most of the investigated cases. We also show that an  $\alpha$ -mixture of the two approaches yields better results than either model separately. In our approach we have assumed that the prediction of the switching variable factorises over the state space. However, one could use a structured output model to predict the joint distribution of different factors. Finally, another issue is the lack of explicit temporal continuity in the  $s$ -chain. Implicitly, this is handled by the feature construction process. However, a future direction could be to establish a Markovian connection on the  $s$ -chain too and compare with our current approach.

#### Acknowledgements

We extend our thanks to Ian Piper and Christopher Hawthorne for their expert insight and annotated data, and to Martin Shaw and Partha Lal for preprocessing code and valuable discussions. Author KG was funded by the Scottish Informatics and Computer Science Alliance. This research was funded in part by the Chief Scientist Office

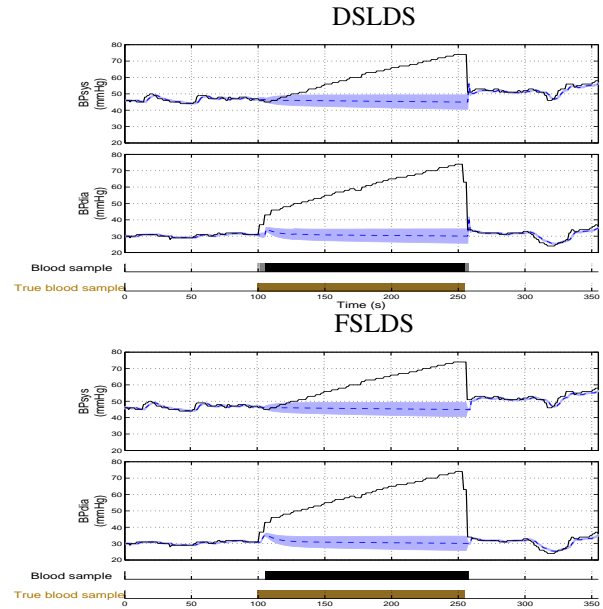


Figure 6: Example of the inferred underlying physiology in the presence of a blood sample in the case of the DSLDS (top) and the FSLDS (bottom). The solid line corresponds to the actual observations, while the estimated true physiology is plotted as a dashed line with the shaded area indicating two standard deviations.

(Scotland) ref. CZH/4/801.

#### References

Akaike, H. (1972). Information theory and an extension of the maximum likelihood principle. *2nd Int. Symp. Information Theory, Supp. to Problems of Control and Information Theory*, pages 267–281.

Alspach, D. L. and Sorenson, H. W. (1972). Nonlinear Bayesian Estimation Using Gaussian Sum Approximations. *Automatic Control, IEEE Transactions on*, **17**(4), 439–448.

Amari, S.-i. (2007). Integration of Stochastic Models by Minimizing  $\alpha$ -Divergence. *Neural Computation*, **19**(10), 2780–2796.

Breiman, L. (2001). Random Forests. *Machine Learning*, **45**(1), 5–32.

Brockwell, P. J. and Davis, R. A. (2009). *Time Series: Theory and Methods*. Springer.

Diggle, P. (1990). *Time Series: A Biostatistical Introduction*. Oxford University Press.

Ghahramani, Z. and Hinton, G. E. (1996). Parameter Estimation for Linear Dynamical Systems. Technical report, Technical Report CRG-TR-96-2, University of Toronto, Dept. of Computer Science.

- Goldberger, A. L., Amaral, L. A., Glass, L., Hausdorff, J. M., Ivanov, P. C., Mark, R. G., Mietus, J. E., Moody, G. B., Peng, C.-K., and Stanley, H. E. (2000). PhysioBank, physioToolkit, and physioNet components of a new research resource for complex physiologic signals. *Circulation*, **101**(23), e215–e220.
- Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning*. Springer.
- Lafferty, J., McCallum, A., and Pereira, F. C. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning (ICML)*.
- Lehman, L., Adams, R., Mayaud, L., Moody, G., Malhotra, A., Mark, R., and Nemati, S. (2014). A physiological time series dynamics-based approach to patient monitoring and outcome prediction. *Biomedical and Health Informatics*.
- Lerner, U. and Parr, R. (2001). Inference in Hybrid Networks: Theoretical Limits and Practical Algorithms. In *Proceedings of the Seventeenth conference on Uncertainty in Artificial Intelligence (UAI)*, pages 310–318. Morgan Kaufmann Publishers Inc.
- Lu, W.-L., Murphy, K. P., Little, J. J., Sheffer, A., and Fu, H. (2009). A Hybrid Conditional Random Field for Estimating the Underlying Ground Surface from Airborne LiDAR Data. *Geoscience and Remote Sensing, IEEE Transactions on*, **47**(8), 2913–2922.
- McCallum, A., Freitag, D., and Pereira, F. C. (2000). Maximum Entropy Markov Models for Information Extraction and Segmentation. In *International Conference on Machine Learning (ICML)*, pages 591–598.
- Murphy, K. P. (1998). Switching Kalman Filters. Technical report, U.C. Berkeley.
- Nemati, S., Lehman, L.-W., and Adams, R. P. (2013). Learning outcome-discriminative dynamics in multivariate physiological cohort time series. In *Engineering in Medicine and Biology Society (EMBC), 2013 35th Annual International Conference of the IEEE*, pages 7104–7107. IEEE.
- Quinn, J. A., Williams, C. K., and McIntosh, N. (2009). Factorial Switching Linear Dynamical Systems applied to Physiological Condition Monitoring. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **31**(9), 1537–1551.
- Särkkä, S. (2013). *Bayesian Filtering and Smoothing*. Cambridge University Press.
- Shumway, R. H. and Stoffer, D. S. (2000). *Time Series Analysis and Its Applications*. Springer New York.
- Stanculescu, I., Williams, C. K., and Freer, Y. (2014). A Hierarchical Switching Linear Dynamical System Applied to the Detection of Sepsis in Neonatal Condition Monitoring. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 752–761. AUAI Press.
- Varma, S. and Simon, R. (2006). Bias in error estimation when using cross-validation for model selection. *BMC Bioinformatics*, **7**(1), 91.
- Williams, C. K. and Stanculescu, I. (2011). Automating the Calibration of a Neonatal Condition Monitoring System. In *Artificial Intelligence in Medicine*, pages 240–249. Springer.

---

# Revisiting Non-Progressive Influence Models: Scalable Influence Maximization in Social Networks

---

**Golshan Golnari \***  
Electrical and Computer  
Engineering Dept.  
University of Minnesota  
Minneapolis, MN 55455

**Amir Asiaee T. \***  
Computer Science and  
Engineering Dept.  
University of Minnesota  
Minneapolis, MN 55455

**Arindam Banerjee**  
Computer Science and  
Engineering Dept.  
University of Minnesota  
Minneapolis, MN 55455

**Zhi-Li Zhang**  
Computer Science and  
Engineering Dept.  
University of Minnesota  
Minneapolis, MN 55455

## Abstract

While influence maximization in social networks has been studied extensively in computer science community for the last decade the focus has been on the *progressive* influence models, such as independent cascade (IC) and Linear threshold (LT) models, which cannot capture the *reversibility of choices*. In this paper, we present the Heat Conduction (HC) model which is a *non-progressive influence model* with real-world interpretations. We show that HC unifies, generalizes, and extends the existing non-progressive models, such as the Voter model [1] and non-progressive LT [2]. We then prove that selecting the optimal seed set of influential nodes is NP-hard for HC but by establishing the submodularity of influence spread, we can tackle the influence maximization problem with a scalable and provably near-optimal greedy algorithm. We are the first to present a scalable solution for influence maximization under non-progressive LT model, as a special case of the HC model. In sharp contrast to the other greedy influence maximization methods, our fast and efficient C2GREEDY algorithm benefits from two analytically computable steps: *closed-form* computation for finding the influence spread as well as the greedy seed selection. Through extensive experiments on several large real and synthetic networks, we show that C2GREEDY outperforms the state-of-the-art methods, in terms of both influence spread and scalability.

## 1 INTRODUCTION

Motivated by viral marketing and other applications, the problem of influence maximization in a social network has attracted much attention in recent years. Given a social network where nodes represent users in a social group, and edges represent relationships and interactions between the

users (and through which they influence each other), the basic idea of influence maximization is to select an initial set of “most influential” users (often referred to as the *seeds*) among all users so as to maximize the total influence under a given diffusion process (often referred to as the *influence model*) on the social network. In the context of viral marketing, this amounts to initially targeting a set of influential customers, e.g., by providing them with free product samples, with the goal to trigger a cascade of influence through “word-of-mouth” or recommendations to friends to maximize the total number of customers adopting the said product. Domingos and Richardson [3] introduced this algorithmic problem to the Computer Science community and Kempe et al. [2] made the topic vastly popular under the name of *influence maximization*. They studied two influence models, the independent cascade (IC) model and the linear threshold (LT) model, and applied a greedy method to tackle the influence maximization problem [2]. Unfortunately Kempe et al.’s approach [2] for calculating the influence spread is based on Monte Carlo simulations which does not scale to large networks [4,5]. As the result, it motivated researchers to either improve the scalability [4,5] or study more tractable influence models [6,7].

The focus of almost all of these earlier studies are, however, *progressive* influence models, including LT and IC models, in which once a customer adopts a product or a user performs an action she cannot revert it. Retweeting news and sharing videos in online social network websites, are examples of progressive, i.e. irreversible actions. Nevertheless, there are numerous real world instances where the actions are *non-progressive*, especially in the technology adoption domain. For example, adopting a cell phone service provider, such as AT&T and T-mobile, is a non-progressive action where a user can switch between providers. The objective of influence maximization in this example is to persuade more users to adopt the intended provider for a longer period of time. To capture the reversibility of choices in real scenarios, we present the Heat Conduction (HC) model that unifies, generalizes, and extends the existing non-progressive models, including non-progressive LT (NLT) [2] and Voter model [1] (see Section 5). In contrast

---

\* G. Golnari and A. Asiaee contributed equally to this work.

to the Voter model, HC does not *necessarily* reach consensus, where one product dominates and extinguishes the others after finite time, so the proposed HC model can explain the *coexistence of multiple product adoptions*, which is a typical phenomena in the real world. In addition, the HC model incorporates both “social” and “non-social” factors, e.g., intrinsic inertia or reluctance of some users in adopting a new idea or trying out a new product, external “media effect” which exerts a “non-social” influence in promoting certain ideas or products.

We tackle the influence maximization problem under the HC influence model with a *scalable* and provably *near-optimal* solution. The approach by Kempe et al. [2] for influence maximization under NLT model, is to reduce the model to (progressive) LT by replicating the network as many as time progresses and compute the influence spread by the same slow Monte Carlo method for the resulted huge network. This approach is practically impossible for large networks, specially for the *infinite time horizon*. We also prove that contrary to the Voter model, for which the influence maximization can be solved *exactly* in polynomial time [1], the influence maximization for HC is NP-hard. We develop an approximation (greedy) algorithm for influence maximization under HC for infinite time horizon with guaranteed *near-optimal* performance. We are able to provide *closed form* solution for both computing the influence spread and greedy selection step which entirely removes the need to explicitly evaluate each node as the best seed candidate. Our fast and scalable algorithm, C2GREEDY, for influence maximization under the HC model removes the computational barrier that prevented the literature from considering the non-progressive influence models.

Our extensive experiments on several large real and synthetic networks validate the efficiency and effectiveness of our method which outperforms the state-of-the-art in terms of both influence spread and scalability. We show that the most influential nodes under progressive models does not necessarily act as the most influential nodes under non-progressive models and a *designated* non-progressive algorithm is necessary. Moreover, we present the first real non-progressive cascade dataset which models the non-progressive propagation of research topics among network of researchers. Our contribution in this paper is as follows:

- We propose the HC influence model which unifies, generalizes, and extends the existing non-progressive models.
- We show that the HC has three key properties which enables us to solve the influence maximization efficiently.
- To the best of our knowledge, we are the first to present a scalable solution for influence maximization under non-progressive LT model.
- We demonstrate high performance and scalability of our algorithm via extensive experiments and present the first ever real non-progressive cascade dataset.

The rest of this paper is organized as follows. After a brief

review on the related work, we introduce our HC model in Section 2. Next, we show how to compute the influence spread for HC in closed form in Section 3. In Section 4, we present our efficient algorithm C2GREEDY for influence maximization under the HC model. Section 5 explains how HC unifies other non-progressive models and provides a more complete view of the HC model. Finally we conduct comprehensive experiments in Section 6 to illustrate performance of our algorithm.

**Related work.** After the debut of influence maximization as a data mining problem [3], it is formulated as a discrete optimization problem based on progressive influence models (LT and IC) from social and physical sciences [2]. Kempe et al. [2] show that influence maximization is NP-hard under LT and IC models but the influence spread is submodular for the models which enables them to use the greedy method. Although the algorithm is greedy it usually does not scale, because it needs to compute influence spread many times in each iteration while influence spread has no known closed form and is estimated by Monte Carlo simulation. The follow-up studies [4–9] attempt to speed up this process by avoiding or decreasing the need for the MC simulation (for further details of the studies on progressive influence model please refer to Supplementary). Kempe et al. [2] also introduce a non-progressive version of the LT influence model (NLT) and try to tackle the influence maximization problem under NLT by reducing the model to (progressive) LT, discussed in Section 1.

The Voter model, as the most well-known non-progressive model, is originally introduced in [10, 11] and adopted for viral marketing in [1]. Even-Dar and Shapira show that under the Voter model, highest degree nodes are the solution of influence maximization [1]. Unfortunately since the Voter model reaches consensus, i.e. one product remains in long term, it can not explain the coexistence of multiple product adoptions, which is a typical case in many real product adoptions.

## 2 HEAT CONDUCTION INFLUENCE MODEL

The heat conduction (HC) influence model is inspired by the resemblance of influence diffusion through a social network to heat conduction through an object, where heat is transferred from the part with higher temperature to the part with lower temperature. We provide a simple description of HC in this section and defer the complete view of it as well as its unification property to Section 5.

Considering a directed graph  $G = (\mathcal{V}, \mathcal{E})$  which represents a social (influence) network, where the directed edge from node  $i$  to node  $j$  declares that  $i$  follows  $j$  (or equivalently  $j$  influences  $i$ ). The edge weight  $\omega_{ij}$  indicates the amount that  $i$  trusts  $j$ , and  $0 \leq \omega_{ij} \leq 1$ . The set of  $i$ 's neighbors, representing the nodes that influence  $i$ , is denoted by  $\mathcal{N}(i)$ . The influence cascade can be assumed as a *binary* process in which a node who adopts the “desired” product is called

*active*, and *inactive* otherwise. Note that this assumption holds for the cases with multiple products as well, where the objective is to maximize the influence (publicity) of the “desired” product, and the rest are all considered “undesired”. *Seed* is a node that has been selected for the direct marketing and remains active during the entire process. In the HC model, the influence cascade is initiated from a set of seeds  $S$  and arbitrary values for other nodes. The *choice* of node  $i$  to become active or inactive at time  $t + 1$  is a linear function of the choices of its neighbors at time  $t$  as well as its intrinsic (or non-social) bias toward activeness:

$$Pr(\delta_i(t+1) = 1 | \mathcal{N}(i)) = \beta_i b + (1 - \beta_i) \sum_{j \in \mathcal{N}(i)} \omega_{ij} \delta_j(t), \quad (1)$$

where  $\beta_i \in (0, 1)$ ,  $b \in [0, 1]$ , and  $\sum_{j \in \mathcal{N}(i)} \omega_{ij} = 1$ . Indicator function  $\delta_i(t)$  is 1 when node  $i$  adopts the desired product at time  $t$  and 0 otherwise. We refer to (1) as the *choice rule*. The dependence on neighbors in (1) represents the “social” influence and the bias value  $b$  accounts for “non-social” influence which comes from any source out of the neighbors, e.g. media. The “non-social” influence can explain the cases where the “social” influence alone fails to model the cascades [12]. We discuss further interpretation and extensions of HC in Section 5.

Replacing the choice rule (1) in  $Pr(\delta_i(t + 1)) = \sum Pr(\delta_i(t + 1) | \mathcal{N}(i)) Pr(\mathcal{N}(i))$  results in the following *probabilistic* interpretation of the original binary HC model. Each node  $i$  has a value at time  $t$  denoted by  $u(i, t)$  which represents the *probability* that she adopts the desired product at time  $t$ :

$$u(i, t + 1) = \beta_i b + (1 - \beta_i) \sum_{j \in \mathcal{N}(i)} \omega_{ij} u(j, t), \quad (2)$$

Simple calculation shows that the bias value  $b$  can be integrated into the network by adding a bias node  $n$  (assuming that the network has  $n - 1$  nodes) with adoption probability  $b$ . Therefore, HC dynamics converts to the following:

$$u(i, t + 1) = \sum_{j \in \mathcal{EN}(i)} P_{ij} u(j, t), \quad (3)$$

where  $\mathcal{EN}(i) = \mathcal{N}(i) \cup \{n\}$  is the extended neighborhood,  $P_{in} = \beta_i$ ,  $u(n, t) = b$ , and  $\forall j \neq n : P_{ij} = (1 - \beta_i) \omega_{ij}$ . Rewriting (3) in the following form shows that HC follows the discrete form of **Heat Equation** [13], which reveals the naming reason of HC influence model:  $u(:, t + 1) - u(:, t) = (P - I)u(:, t)$ , where  $\mathcal{L} = I - P$  is the Laplacian matrix,  $u(i, t)$  is the temperature of particle  $i$  at time  $t$ , and “:” denotes the vector of all entries.

### 3 HC INFLUENCE SPREAD

The influence spread of set  $S$  for time  $t$  is defined as the expected number of active nodes at time  $t$  of a cascade started with  $S$ . Knowing that  $u(i, t)$  is the probability of node  $i$  being active at time  $t$ , *influence spread* (or function)  $\sigma(\mathcal{S}, t)$

is computed from:

$$\sigma(\mathcal{S}, t) = \sum_{i \in \mathcal{V}} u(i, t). \quad (4)$$

Motivated by the classical heat transfer methods, the initial and the boundary conditions should be specified to solve the heat equation and find  $u(i, t)$  uniquely. In HC, the seeds  $S$  and the bias node are the boundary nodes and the rest are interiors. Assuming  $\mathcal{S} = \{n - 1, n - 2, \dots, n - |\mathcal{S}|\}$  and  $n$  as the bias node, HC is defined by the following heat equation system:

$$\begin{aligned} \text{Main equation} & : u(:, t + 1) - u(:, t) = -\mathcal{L}u(:, t) \\ \text{Boundary conditions} & : u(n, t) = b, \\ & u(s, t) = 1 \quad \forall s \in \mathcal{S} \\ \text{Initial condition} & : u(:, 0) = z + [0, \dots, 0, \underbrace{1, \dots, 1}_{|\mathcal{S}|}, b]', \end{aligned} \quad (5)$$

where, as indicated in this formula, initial value  $u(:, 0)$  is the sum of two vectors: the initial values of the interior nodes ( $z$ ) and the initial values of boundaries (the second vector). The corresponding entries of boundaries in  $z$  are zero. In the continue, exploiting probability theory and novel Markov chain metrics, we provide a closed form solution to this heat equation system.

Social network  $G$  can be interpreted as an absorbing Markov chain where the absorbing states (boundary set  $\mathcal{B}$ ) are the seeds and bias node,  $\mathcal{B} = \mathcal{S} \cup \{n\}$ , and  $P_{ij}$  is the probability of transition from  $i$  to  $j$ . The adoption probability of the nodes at time  $t$ , i.e.  $u(:, t)$ , can be written as a linear function of initial condition (3):

$$u(:, t) = P^t u(:, 0), \quad (6)$$

where  $P$  is row-stochastic and has the following block form:  $P = \begin{bmatrix} R & B \\ \mathbf{0} & I \end{bmatrix}$ . The superscript indicates the time here. The boundary set by definition have fixed values over time and do not follow any other nodes which leads to the zero and identity blocks  $I_{(|\mathcal{S}+1|) \times (|\mathcal{S}+1|)}$ . Blocks  $R$  and  $B$  represent transition probabilities of interior-to-interior and interior-to-boundary respectively. Note that different boundary conditions in (5), like different seed set, result in a different  $P$ . Therefore both  $P$  and  $u(:, t)$  implicitly depend on  $\mathcal{S}$ .

When  $t$  goes to infinity, transient part of  $u$  vanishes and it converges to the steady-state solution  $v = u(:, \infty)$ , which is independent of time and is Harmonic, meaning that it satisfies  $Pv = v$  [14]. Assume  $v = (v_{\mathcal{I}}, v_{\mathcal{B}})^T$  where  $\mathcal{I} = \mathcal{V} \setminus \mathcal{B}$  is the set of interior nodes, then the value of interior nodes is computed from boundary nodes [14]:

$$v_{\mathcal{I}} = (I - R)^{-1} B v_{\mathcal{B}} = F B v_{\mathcal{B}} = Q v_{\mathcal{B}}. \quad (7)$$

where  $F = (I - R)^{-1}$  is the *fundamental matrix* and  $F_{ij}$  indicates the average number of times that a random walk



started from  $i$  passes  $j$  before absorption by any absorbing (boundary) nodes [14]. Also, the *absorption probability* matrix  $Q = FB$  is a  $(n - |\mathcal{S}| - 1) \times (|\mathcal{S}| + 1)$  row-stochastic matrix, where  $Q_{ij}$  denotes the probability of absorption of a random walk started from  $i$  by the absorbing node  $j$  [14].

From here on, without loss of generality, we assume  $b$  to be zero in equation (5). Using (6) and (7), the influence spreads for infinite time can be computed in closed form:

$$\sigma(\mathcal{S}, \infty) = \sum_{i=1}^n v(i) = |\mathcal{S}| + \sum_{i \in \mathcal{I}} \sum_{s \in \mathcal{S}} Q_{is}^{\mathcal{S}}. \quad (8)$$

The superscript in  $Q^{\mathcal{S}}$  and  $P^{\mathcal{S}}$  explicitly indicates that they are functions of seed set  $\mathcal{S}$ . Note that in fact they are depending on the total boundary set,  $\mathcal{B} = \mathcal{S} \cup \{n\}$ , but since the bias node is always a boundary, throughout this paper we discard it from the superscripts to avoid clutter.

## 4 INFLUENCE MAXIMIZATION FOR HC

In this section we solve the influence maximization problem for *infinite time horizon* under the HC model:

$$\mathcal{S}^* = \operatorname{argmax}_{\mathcal{S} \subseteq \mathcal{V}} \sigma(\mathcal{S}, \infty), \quad s.t. \quad |\mathcal{S}| \leq K. \quad (9)$$

### 4.1 INFLUENCE MAXIMIZATION FOR $K = 1$

Based on (8) and (9), the most influential person (MIP) is the solution of the following optimization problem:  $\operatorname{argmax}_{\mathcal{V} \setminus \{n\}} \sum_{i \in \mathcal{V} \setminus \{s, n\}} Q_{is}^{\{s\}}$ . This equation states that to find the MIP, we need to pick each candidate  $s$  and make it absorbing and compute the new  $P$  as  $P^{\{s\}}$  which in turn changes  $Q$  to  $Q^{\{s\}}$ , and repeat this procedure  $n - 1$  times for all  $s$ . This procedure is problematic because for each  $Q^{\{s\}}$  we require to recompute matrix  $F^{\{s\}}$  which involves matrix inversion. But, in the following theorem we show that we are able to do this by only one matrix inversion instead of  $n - 1$  matrix inversions, and having matrix  $F^{\emptyset}$  is enough to find the most influential person of the network ( $\emptyset$  sign indicated no seed is selected):

**Theorem 1.** *MIP under HC (1) when  $t \rightarrow \infty$  can be computed in closed form from the following formula:*

$$MIP = \operatorname{argmax}_{s \in \mathcal{V} \setminus \{n\}} \sum_{i \in \mathcal{V} \setminus \{n\}} \frac{F_{is}^{\emptyset}}{F_{ss}^{\emptyset}} = \operatorname{argmax} \mathbf{1}' \check{F}^{\emptyset}, \quad (10)$$

where  $\check{F}^{\emptyset}$  is  $F^{\emptyset}$  when each of its columns is normalized by the corresponding diagonal entry. Note that left multiplication of all ones row vector is just a column-sum operation.

### 4.2 INFLUENCE MAXIMIZATION FOR $K > 1$

Although the influence maximization can be solved optimally for  $K = 1$ , the general problem (9) under HC for  $K > 1$  is NP-hard:

**Theorem 2.** *Given a network  $G = (\mathcal{V}, \mathcal{E})$  and a seed set  $\mathcal{S} \subseteq \mathcal{V}$ , influence maximization for infinite time horizon (9) under HC defined by (1) is NP-hard.*

In spite of being NP-hard, we show that the influence spread  $\sigma(\mathcal{S}, \infty)$  is *submodular* in the seed set  $\mathcal{S}$  which enables us to find a provable near-optimal greedy solution. A set function  $f : 2^{\mathcal{V}} \rightarrow \mathbb{R}$  maps subsets of a finite set  $\mathcal{V}$  to the real numbers and is submodular if for  $\mathcal{T} \subseteq \mathcal{S} \subseteq \mathcal{V}$  and  $s \in \mathcal{V} \setminus \mathcal{S}$ ,  $f(\mathcal{T} \cup \{s\}) - f(\mathcal{T}) \geq f(\mathcal{S} \cup \{s\}) - f(\mathcal{S})$  holds, which is the diminishing return property. Following theorem presents our established submodularity results.

**Theorem 3.** *Given a network  $G = (\mathcal{V}, \mathcal{E})$ , influence spread  $\sigma(\mathcal{S}, \infty)$  under the HC model is non-negative monotone submodular function.*

The greedy solution adds nodes to the seed set  $\mathcal{S}$  sequentially and maximizes a monotone submodular function with  $(1 - 1/e)$  factor approximation guarantee [15]. More formally the  $(k + 1)$ -th seed is the node with maximum **marginal gain**:  $(k + 1)$ -th-MIP $_t = \operatorname{argmax}_{s \in \mathcal{V} \setminus \{\mathcal{S}_k \cup \{n\}\}} \sigma(\mathcal{S}_k \cup \{s\}, t) - \sigma(\mathcal{S}_k, t)$ , where  $\mathcal{S}_k$  is the set of  $k$  seeds which have been picked already. Although we can compute the above objective function in closed form, for selecting the next seed we have to test all  $s$  to solve the problem which is the approach of all existing greedy based method in the literature. Previously a lazy greedy scheme have been introduced to reduce the number testing candidate nodes  $s$  [8]. In the next section we go one step further and show that under the HC model and for *infinite time horizon* we can solve the marginal gain in *closed form*.

### 4.3 GREEDY SELECTION

An important characteristic of the linear systems, like HC when  $t \rightarrow \infty$ , is the ‘‘superposition’’ principle. We leverage this principle to calculate the marginal gain of the nodes efficiently and pick the one with maximum gain for the greedy algorithm. Based on this principle, the value of each node in HC for infinite time, and for a given seed set  $\mathcal{S}$ , is equal to the algebraic sum of the values caused by each seed acting alone, while all other values of seeds have been kept zero. Therefore, when a node  $s$  is added to the seed set  $\mathcal{S}_k$ , its marginal gain can be calculated as the summation of values of the nodes when all of the values of  $\mathcal{S}_k$  have been turned to zero and node  $s$  is the only seed in the network, whose value is  $1 - v^{\mathcal{S}_k}(s)$ . In this new problem, the vector of boundary values  $v_{\mathcal{B}}^{\mathcal{S}_k \cup \{s\}}$  is a vector of all 0’s except the entry corresponding to the node  $s$  with value  $1 - v^{\mathcal{S}_k}(s)$ , and the value of interior node  $i$  is obtained from (7):

$$v_{\mathcal{I}}^{\mathcal{S}_k \cup \{s\}}(i) = Q_{is}^{\mathcal{S}_k \cup \{s\}}(1 - v^{\mathcal{S}_k}(s))$$

Substituting  $Q$  from lemma 3 result (see Supplementary), the  $k + 1$ -th seed is determined from the following closed

form equation:

$$\begin{aligned}
& (k+1)\text{th-MIP} \\
& = \operatorname{argmax}_{s \in \mathcal{V} \setminus \{\mathcal{S}_k \cup \{n\}\}} \sum_{i \in \mathcal{V} \setminus \{\mathcal{S}_k \cup \{n\}\}} \frac{F_{is}^{S_k}}{F_{ss}^{S_k}} (1 - v^{S_k}(s)), \\
& = \operatorname{argmax}(\mathbf{1} - v^{S_k})' \check{F}^{S_k} \tag{11}
\end{aligned}$$

Note that vector  $v^{S_k}$  is obtained in step  $k$  and is known, and matrix  $F^{S_k}$  can be calculated from  $F^{S_{k-1}}$  without any need for matrix inversion (see Supplementary, lemma 1). One may observe that equation (11) is the general form of Theorem 1, since  $v^{S_0} = v^\emptyset = 0$ . Notice that equation (11) intuitively uses two criteria for selecting the new seed: its current value should be far from 1 (higher value for  $(1 - v^{S_k}(s))$  term) which suggests that it is far from the previously selected seeds, and at the same time it should have a high network centrality (corresponding to the  $F_{is}^{S_k}/F_{ss}^{S_k}$  term). Algorithm 1 summarizes our C2GREEDY method for  $t \rightarrow \infty$ : a greedy algorithm with 2 closed form steps. Operator  $\otimes$  in step 10 denotes the Hadamard product.

---

#### Algorithm 1 C2GREEDY

---

- 1: **input:** extended directed network  $G = (\mathcal{V}, \mathcal{E})$  with bias node  $n$ , maximum budget  $K$ .
  - 2: **output:** seed set  $\mathcal{S}_K \subseteq \mathcal{V}$  with cardinality  $K$ .
  - 3: compute matrix  $P$  from  $G$ .
  - 4:  $\mathcal{S}_0 := \emptyset$
  - 5:  $F^{S_0} := (I - P^{S_0})^{-1}$
  - 6:  $s = \operatorname{argmax} \mathbf{1}' \check{F}^\emptyset$ , and  $\mathcal{S}_1 = \mathcal{S}_0 \cup \{s\}$
  - 7:  $v^{S_1} = \check{F}^{S_0}(:, s)$
  - 8: **for**  $k = 1$  to  $K - 1$  **do**
  - 9:  $\forall i, j \in \mathcal{I} : F_{ij}^{S_k \cup \{s\}} = F_{ij}^{S_k} - \frac{F_{is}^{S_k} F_{sj}^{S_k}}{F_{ss}^{S_k}}$
  - 10:  $s = \operatorname{argmax}(\mathbf{1} - v^{S_k})' \otimes \mathbf{1}' \check{F}^{S_k}$ , and  $\mathcal{S}_{k+1} = \mathcal{S}_k \cup \{s\}$
  - 11:  $v^{S_{k+1}} = v^{S_k} + (\mathbf{1} - v^{S_k}(s)) \check{F}^{S_k}(:, s)$
  - 12: **end for**
- 

## 5 DISCUSSION

In this section, we present the comprehensive view of the HC model and elaborate its (unifying) relation to the other models by providing multiple interpretations.

**Social interpretation.** HC can be simply extended to model many real cases that the other influence models fail to cover. As briefly mentioned in Section 2, the original HC (1), models both “social” and “non-social” influences which cover the observations from the real datasets [12]. The extension of HC which is more flexible in modeling real world cascades is as follows:

$$u(i, t+1) = m\alpha_i + r\gamma_i + (1 - \gamma_i - \alpha_i) \sum_{j \in \mathcal{N}(i)} \omega_{ij} u(j, t), \tag{12}$$

where,  $\sum_{j \in \mathcal{N}(i)} \omega_{ij} = 1$ ,  $\gamma_i, \alpha_i \in [0, 1]$ ,  $m = 1$ , and  $r = 0$ . Factor  $r$  models the “discouraging” factor like intrinsic *reluctance* of customers toward a new product, and  $m$  represents “encouraging” factor like *media* that promotes the new product. These two factors can explain cases

where all neighbors of a node are active but the node remains inactive, or when a node becomes active while none of her neighbors are active [12]. Note that all of the formulas and results stated so far is simply applicable to the general HC model (12).

**Unification of existing non-progressive models.** HC (1) unifies and extends many of the existing non-progressive models. In the Voter model, a node updates its choice at each time step by picking one of its neighbors randomly and adopting its choice. In other words, the choice rule of node  $i$  is the ratio of the number of her active neighbors to her total number of neighbors. Thus, Voter’s choice rule is the simplified form of HC’s choice rule (1) where  $\omega_{ij}$  is equal to  $\frac{1}{d_i}$  ( $d_i$  is the out-degree of node  $i$ ) and all  $\beta_i$ s are set to zero. Also, note that having  $\beta_i = 0$  indicates that the Voter does not cover the “non-social” influence.

In the *non-progressive* LT (NLT) [2], each node is assigned a random threshold  $\theta$  at each time step and becomes active if the weighted number of its active neighbors (at previous time step) becomes larger than its threshold:  $\sum_{j \in \mathcal{N}(i)} \omega_{ij} \delta_j(t) \geq \theta_i(t+1)$ , where the edge weights satisfy  $\sum_{j \in \mathcal{N}(i)} \omega_{ij} \leq 1$ . Thus, the choice rule of node  $i$  at time  $(t+1)$  under the NLT is obtained from the following equation:

$$\begin{aligned}
Pr(\delta_i(t+1) = 1 | \mathcal{N}(i)) &= Pr(\theta_i(t+1) \leq \sum_{j \in \mathcal{N}(i)} \omega_{ij}^{\text{NLT}} \delta_j(t)) \\
&= \sum_{j \in \mathcal{N}(i)} \omega_{ij}^{\text{NLT}} \delta_j(t), \tag{13}
\end{aligned}$$

where the second equality is the result of sampling  $\theta_i(t+1)$  from the *uniform distribution*  $U(0, 1)$ . Equation (13) is the simplified form of HC’s choice rule (1), where  $b = 0$  and  $(1 - \beta_i) \omega_{ij}^{\text{HC}} = \omega_{ij}^{\text{NLT}}$ . Note that since in the NLT  $b$  accepts only zero value, this influence model also cannot cover *encouraging* “non-social” influence. Moreover, if the edge weights’ gap in NLT, i.e.  $g_i = 1 - \sum_{j \in \mathcal{N}(i)} \omega_{ij}^{\text{NLT}}$ , is zero for all the nodes, it cannot model the “non-social” influence at all, since the corresponding  $\beta_i$ ’s in (1) would be equal to zero in that case.

Generalized linear threshold (GLT) is another non-progressive model proposed in [16] to model the adoption process of *multiple* products. Assigning a color  $c \in \mathcal{C}$  to each product, a node updates its color, at each time step, by randomly picking one of its neighbors based on its edge weights and adopts the selected neighbor’s color. For binary case  $|\mathcal{C}| = 2$ , where we only distinct between adoption of a desired product (active) and the rest of products (inactive), GLT’s choice rule reduces to the following equation:  $Pr(\delta_i(t+1) = 1 | \mathcal{N}(i)) = \frac{\beta}{2} + (1 - \beta) \sum_{j \in \mathcal{N}(i)} \omega_{ij} \delta_j(t)$ . It is easy to see that this is the restricted form of HC’s choice rule (1), where nodes are all connected to the bias node with equal weight of  $\beta$  and bias value  $b$  has to be  $\frac{\beta}{2}$ .

**Physical interpretation.** We showed that the existing non-progressive models are special cases of HC, and in this part we describe their equal heat conduction system

Table 1: Specifying the equal heat system for existing non-progressive influence models.

| Model | Non-Social influence | Weighted edges | Boundary     |             | Init. Cond. |          | Equivalent Physical Heat Conduction System    |
|-------|----------------------|----------------|--------------|-------------|-------------|----------|---|
|       |                      |                | High $T = 1$ | Low $T < 1$ | $= 0$       | $\neq 0$ |   |
| NLT1  | ✓                    | ✓              |              | ✓           | ✓           |          | Circular ring with a fixed-temp. point        |
| NLT2  | ✓                    | ✓              | ✓            | ✓           | ✓           |          | A rod with fixed-temp. ends, one high one low |
| NLT3  |                      | ✓              |              |             | ✓           |          | (Isolated) circular ring                      |
| NLT4  |                      | ✓              | ✓            |             | ✓           |          | Circular ring with a fixed-temp. point        |
| Voter |                      |                |              |             |             | ✓        | (Isolated) circular ring                      |
| GLT   | ✓                    | ✓              |              | ✓           |             |          | Circular ring with a fixed-temp. point        |

which are uniquely specified by the initial and boundary conditions. Table 1 summarizes the heat interpretation of the influence models. We introduce four variants of non-progressive LT, based on two factors: seed and gap  $g_i$ . NLT1 and NLT2 support non-zero gaps, and NLT2 and NLT4 allows seeds, i.e. nodes in the network that always remain active. The non-progressive LT model presented in [2] is equivalent to NLT2. Reluctance factor and seeds in all models are equivalent to the low and high temperature boundaries respectively, and initial condition addresses the interiors’ initial values ( $z$  in (5)). The non-social influence and edge weights factors appear in the Laplacian matrix calculation of (5). The equivalent physical heat conduction systems are easy to understand, here we just briefly point out the equivalence of the Voter model and the isolated circular ring. Circular ring is a rod whose ends are connected to each other and do not have any energy exchange with outside [17] which explains why the Voter conserves the total initial heat energy, and reaches to an equilibrium with an equal temperature for all of the nodes, i.e., consensus.

**Random walk interpretation.** Beside the heat conduction view, the random walk prospect helps to gain a better understanding of the models and their relations. Assume that active and inactive nodes are colored black and white respectively. Consider the *original view* of any influence model which is the actual process that unfolds in time, so we look at the time-forward direction. We take a snapshot of the colored network at each time step  $t$ . Putting together the sequence of snapshots, the result is a random walk in the “colored graphs” state space with  $2^n$  states. On the other hand, the *dual view* looks at the time-reverse direction of influence models. It is known for both IC-based models (like IC [2] and ConTinEst [7]) and LT-based models (Table 1 as well as HC and LT) that a single node from  $\mathcal{N}(i)$  is responsible for  $i$ ’s color switch, which we name it as the parent of  $i$ . Now assuming that the process has advanced up to the time  $t$ , we reverse the process by starting from each node  $i$  and follow its ancestors. Here is the point where IC and LT based models separate from each other: due to  $\sum_{j \in \mathcal{N}(i)} \omega_{ij} \leq 1$  constraint, ancestors of  $i$  in the LT-based models form a random walk starting from node  $i$ , which is not the case in IC-based models. Note that we have  $n$  random walks that can meet and merge, thus they are known as *coalescing random walks* [18]. This view also helps us to demonstrate the essential difference between progressive

Table 2: List of networks used in experiments.

|                    |            | $ \mathcal{V} $ | $ \mathcal{E} $    | Params               |
|--------------------|------------|-----------------|--------------------|----------------------|
| Synthetic Networks | Random     | 1024            | -                  | [0.5, 0.5; 0.5, 0.5] |
|                    | Hier.      | 1024            | -                  | [0.9, 0.1; 0.1; 0.9] |
|                    | Core.      | 1024            | -                  | [0.9, 0.5; 0.5, 0.3] |
|                    | ForestFire | 1 – 300K        | $2.5 \mathcal{V} $ | [0.35, 0.25]         |
| Real Networks      | KClub      | 34              | 501                | -                    |
|                    | PBlogs     | 1490            | 19087              | -                    |
|                    | WikiVote   | 7115            | 103689             | -                    |
|                    | MLFWF      | 10604           | 168918             | -                    |

and non-progressive models. Dual view of progressive LT model is a *coalescing self-avoiding walks* which is the outcome of randomizing the threshold  $\theta$  only once at the beginning of the process for the nodes in each realization. This bounds the number of “live” edges [2] connected to each node by one which prevents the creation of “loop” in the influence paths. Note that both counting and finding the probability of self-avoiding walks are  $\#P$  hard [4].

## 6 EXPERIMENTS

In this section, we examine several aspects of C2GREEDY and compare it with state-of-the-art methods. Experiments mainly focus on influence maximization and timing aspects. Finally, we present one example of real non-progressive data and illustrate the result of C2GREEDY.

### 6.1 DATASET

Table 2 summarizes the statistics of the networks that we use throughout the experiments. We work with both synthetic and real networks which we briefly discuss next.

**Synthetic network generation.** We consider the following types of Kronecker network for extensive performance comparison of our method with the state-of-the-art methods: random [19] (parameter matrix [0.5, 0.5; 0.5, 0.5]), hierarchical [20] ([0.9, 0.1; 0.1; 0.9]), and core-periphery [21] ([0.9, 0.5; 0.5, 0.3]). We generate 10 samples from each network and report the average performance of each method. Edge weights are drawn uniformly at random from [0, 1] and weights of each node’s outgoing edges is normalized to 1. For timing experiment, we use ForestFire [20] (Scale-free) network with forward and backward burning probability of 0.35 and 0.25, respectively, and set the outgoing edge weights of node  $i$  to  $1/|\mathcal{N}(i)|$ . The expected density, i.e., number of edges per node, for the resulted ForestFire networks is 2.5.

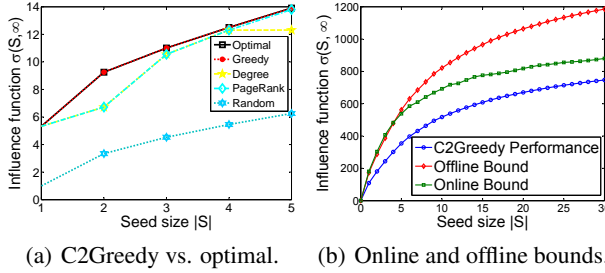


Figure 1: For small network (a) shows C2Greedy matches the optimal performance. For a larger network (b) compares performance of C2Greedy with online and offline bounds.

**Real Networks.** Zachary’s karate club network (KClub) is a small friendship network with 34 nodes and 501 edges [22]. The political blogs network (P Blogs) [23], is a moderate size directed network of hyperlinks between weblogs on US politics with 1490 nodes and 19087 edges. Wikipedia vote network (WikiVote), is the network of who-vote-whom from wikipedia administrator elections [24] with 7115 nodes and 103689 edges. Finally, MLFWF is the network of who-follow-whom in the machine learning research community which we extract from citation networks of combined ACM and DBLP citation network which is available as a part of ArnetMiner [25]. For more information about MLFWF refer to Section 6.4.

For all synthetic and real networks, after constructing the network, we add the bias node to the network and connect all nodes to it with weight  $\beta_i = 0.1$  and re-normalize the weight of the other edges accordingly.

## 6.2 INFLUENCE MAXIMIZATION

In this section we investigate the performance of C2GREEDY in the main task of influence maximization i.e., solving the set function optimization (9). Since finding the optimal solution for (9) is NP-hard, we compare C2GREEDY with optimal solution only for a small network, then for a large network we show that C2GREEDY result is close to the online bound [8]. We also compare the performance of C2GREEDY with the state-of-the-art methods proposed for solving (9) under different (mostly progressive) influence models.

**C2GREEDY vs. optimal.** For testing the quality of C2GREEDY method, we compare its performance with the best seed set (determined by brute force) on a small size network. We work with the KClub network for the brute-force experiment with  $K = 5$ . As Figure 1(a) shows C2GREEDY selects nodes that match the performance of the optimal seed set. In the next step, on a larger network, we show that the performance of C2GREEDY is close to the known online upper bound [8]. We compute the online and offline bounds of greedy influence maximization [8] with  $K = 30$  for P Blogs network. Figure 1(b) illustrates that C2GREEDY result is close to the online bound and therefore close to the optimal solution’s performance.

**C2GREEDY vs. state-of-the-art.** Next, we compare C2GREEDY with the state-of-the-art methods of influence maximization over three aforementioned synthetic networks and WikiVote real network. Among baseline methods PMIA [5] and LDAG [4] are approximation for IC and LT models respectively and SP1M [26] is a shortest-path based heuristic algorithm for influence maximization under IC. ConTinEst [27] is a recent method for solving continuous time model of [6] and PageRank is the well-known information retrieval algorithm [28]. Finally, Degree selects the nodes with highest degree as the most influential and Random picks the seed set randomly.

The comparison results are depicted in Figure 2. Interestingly, our algorithm outperformed all of the baselines. Strangely, ConTinEst performs close to Random (except in the random network). A closer look at the results for three synthetic networks reveal that except ConTinEst’s odd behavior all other methods have persistence rank in performance. C2GREEDY is the best method and is followed by PMIA and LDAG, both in second place, which are closely followed by SP1M. PageRank, Degree and Random are next methods in order. In WikiVote real network of Figure 2(d) surprisingly most of the state-of-the-art methods perform terribly poor and Degree (as the KMIP solution to the Voter model) is the only competitor of C2GREEDY. Result of experiment with WikiVote shows that most influential nodes in a progressive models are not necessary influential in non-progressive ones, and designing non-progressive-specific algorithms (like C2GREEDY) is required for influence maximization under non-progressive models.

## 6.3 SPEED AND SCALABILITY

In this part we illustrate the speed benefits of having two closed form updates in the greedy algorithm and also deal with the required single inverse computation of C2GREEDY to prove the scalability of our method.

**Closed form benefits.** As discussed in Section 4, our main algorithm C2GREEDY benefits from closed form computation for both influence spread (8) and greedy selection (11). To show the gain of these closed form solutions, we run the greedy algorithm in three different settings. First without using any of (8) and (11) which we call GREEDY and uses Monte Carlo simulation to estimate the influence spread. Second we only use (8) to have the closed form for influence spread without closed form greedy update of (11) which results in C1GREEDY, and finally C2GREEDY which uses both (8) and (11). Note that we can add lazy update of [8] (see Supplementary) to GREEDY and C1GREEDY to get LGREEDY and LC1GREEDY respectively. Finally we include the original greedy method [2] of solving LT model (progressive version of our model) and its lazy variant, with 100 iteration of Monte Carlo simulation. Note that for having a good approximation of influence spread in LT model, simulations are run for several thousand iterations, but here we just want to illustrate that

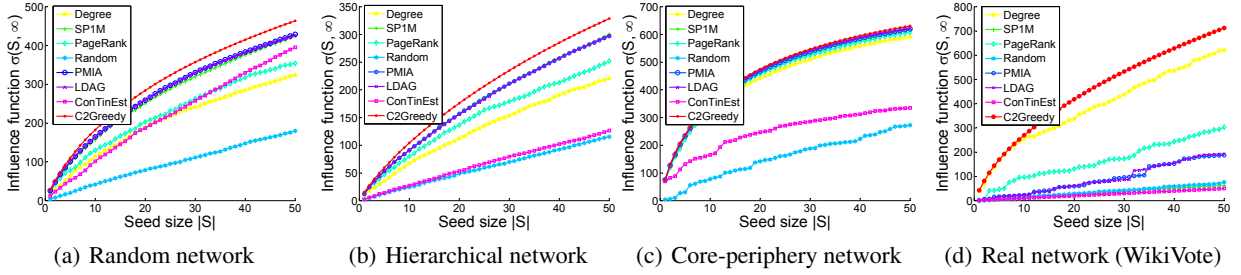


Figure 2: Comparing performance of C2Greedy with state-of-the-art influence maximization methods. Networks of (a), (b), and (c) are synthetic and (d) is a real network.

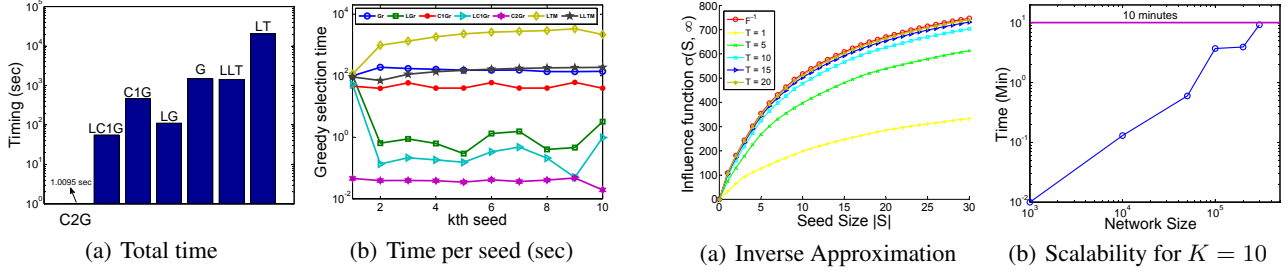


Figure 3: In (a) we compare the total timing of seven algorithms to investigate the effect of closed updates on speed and in (b) we show the per-seed required time for the same experiment.

Figure 4: Timing for inf. max. in large scale networks by exploiting (a) inverse approximation and (b) parallel programming. Results of (b) are on FF networks with edge density 2.5.

the greedy algorithm for HC is much faster than LT, for which 100 iterations is enough. Figure 3(a) illustrates the speed in log-scale of all seven algorithms for  $K = 10$  over the Pblogs dataset [23]. Note that the required time of inverse computation (7) is also included. The results confirm that both closed forms decrease the timing *significantly* (1 sec vs. 461 sec for the next best variation) and help the greedy algorithm far more than the lazy update.

**Per-seed selection time.** The major computational bottleneck of our algorithm is the inverse computation of (7). But fortunately this is needed once and at the beginning of the process. Here assuming offline inverse computation, we are interested in the cost of adding each seed. Figure 3(b) compares the cost of selecting  $k$ -th seed for the five variation of our algorithm, plus LT and LazyLT all described previously. As expected C2GREEDY requires the lowest computation time per seed. Also, the timing per seed for C2GREEDY is strictly decreasing over the size of  $\mathcal{S}$ , because the matrix  $N$  shrinks, while per seed selection time of LT is increasing on average, because more seeds probably lead to bigger cascades.

**Inverse approximation.** Going beyond networks of size  $10^4$  makes the inverse computation problematic, but fortunately we have a good approximation of the inverse through the following expansion:  $F = (I - R)^{-1} \approx I + R^1 + R^2 + \dots + R^T$ . Since all eigenvalues of  $R$  are less than or equal to 1 contribution of  $(R)^i$  to the summation drops very fast as  $i$  increases. The question is how many terms of the expansion,  $T$ , is enough for our application. Heuristically we choose the (effective) diameter of the graph as the number that provides us with a good approximation of  $F^{-1}$ . Note

that the  $i$ th term of the expansion pertains to the shortest paths of size  $i$  between any pair of nodes. Since the graph diameter is the longest shortest path between any pair of nodes, having that many terms gives us a good approximation of  $F^{-1}$ . This is also demonstrated by the experimental result of Figure 4(a) where we compare the result of the influence maximization on the WikiVote network with diameter 15, with actual  $F^{-1}$  and its approximation for different  $T$ 's. As discussed when  $T$  reaches to the diameter, the result of the algorithm that uses inverse approximation coincides with the algorithm that uses the exact inverse.

**Scalability.** Finally to show the scalability of C2GREEDY we perform influence maximization on networks with sizes up to  $3 \times 10^5$ . For speeding up the large scale matrix computation of the Algorithm 1 we developed an MPI version of our code which allows us to run C2GREEDY on computing clusters. Figure 4(b) shows the running time of C2GREEDY for ForestFire networks of sizes varying between 1K to 300K with edge density 2.5 (i.e. ratio of edges to nodes) and effective diameter of 10. The MPI code was run on up to 400 cores of 2.8 GHz. As Figure 4(b) indicates even for the largest tested network with 0.3 million nodes and 0.75 million edges C2GREEDY takes less than 10 minutes for  $K = 10$ .

To give a sense of our achievement in scalability we briefly mention the result of one of the state-of-the-art methods: The scalable ConTinEst [7] runs with 192 cores for almost 60 minutes on ForestFire network of size 100K and edge density of 1.5 to select 10 seeds, where our C2GREEDY finishes in less than 2 minutes for the similar ForestFire

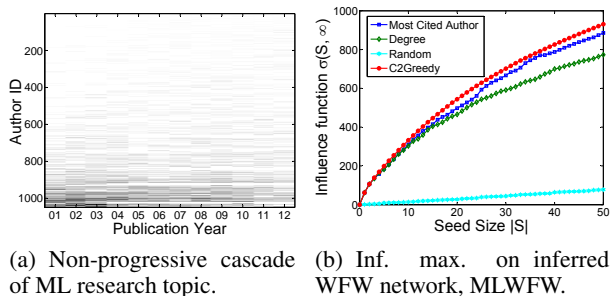


Figure 5: In (a) we show the existence of non-progressive cascade of ML research topic where white means all papers of the author is about ML. In (b) we compare C2Greedy result with other baselines such as most cited author.

network (100K nodes and density 1.8) with 200 cores.

#### 6.4 REAL NON-PROGRESSIVE CASCADE

Collaboration and citation networks are two well-known real networks that have been studied in social network analysis literature [2, 29]. Here we introduce a new network that represents who-follows-whom (WFW) in a research community. Note that the nodes in the collaboration and citation networks are authors and papers respectively but in WFW network nodes are authors and edges are inferred from citations. A directed edges  $(u, v)$  means that author  $u$  has cited one of the papers of author  $v$  which reveals that  $u$  follows/reads papers of  $v$ . Here we investigate the “research topic adoption” cascade. Researchers adopt new research topics during their careers and influence their peers along different research communities. The process starts with an arbitrary research topic for each author and they are influenced by the research topic of those they follow and switch to another topic. For example a data mining researcher that follows mostly the papers of machine learning authors is probably going to switch his research topic to machine learning.

For illustration, we consider only the authors who have published papers in Machine Learning (ML) conferences and journals in a given time period. For the list of ML related conferences and journal we use resources of Arnet-Miner project [25]. We consider each time step a year and study the years 2001 - 2012. An author is an *active* ML author in a given year if at least half of his publications in that year was published in ML venues. Figure 5(a) shows the change in the percentage of ML publication of ML authors who has more than 70 publication in years between 2001 and 2012. As Figure 5(a) suggests, cascade of ML research topic is a non-progressive process and researcher switch back and forth between ML and other alternatives. Among 1049 authors of Figure 5(a) about 400 of them are core ML authors who have rarely published in any other topic, but the non-progressive nature of the process is more visible in the rest (bottom part of Figure 5(a)).

Next we perform influence maximization on the inferred WFW network which we call MLFW network. We ex-

tract the MLFW network from the combined citation network of DBLP and ACM which is publicly available as a part of ArnetMiner project [25] and learn the edge weights similar to the weighted cascade model of [2]. The MLFW network of 2001 - 2012 time frame consists of 10604 authors and 168918 edges. Figure 5(b) compares the result of influence maximization using C2GREEDY and other baselines. Note that other than regular baselines in this specific domain we have another well-known method which is “most cited author” that is equal to selecting authors with highest weighted in-degree in MLFW network. As Figure 5(b) illustrates, C2GREEDY outperforms all of the other methods. Note that the list of  $K$  most influential authors in this experiment means that “if” those authors were switching to the ML topic completely (becoming a member of seed set  $S$ ) they would make the topic vastly popular. Therefore, although the seed set contains the familiar names of well-known ML authors (e.g., Michael I. Jordan and John Lafferty in first and second places), sometimes we encounter exceptions. For example, in the list of top 10 authors selected by C2GREEDY we have “Emery N. Brown” who is a renowned neuroscientist with publications in “Neural Computation” journal.

## 7 CONCLUSION

We introduced the Heat Conduction Model which is able to capture both social influence and non-social influence, and extends many of the existing non-progressive models. We also presented a scalable and provably near-optimal solution for influence maximization problem by establishing three essential properties of HC: 1) submodularity of influence spread, 2) closed form computation for influence spread, and 3) closed form greedy selection. We conducted extensive experiments on networks with hundreds of thousands of nodes and close to million edges where our proposed method gets done in a few minutes, in sharp contrast with the existing methods. The experiments also certified that our method outperforms the state-of-the-art in terms of both influence spread and scalability. Moreover, we exhibited the first real non-progressive cascade dataset for influence maximization. We believe that our method removes the computational barrier that prevented the literature from considering the non-progressive influence models. Studying other forms of non-progressive influence models, such as non-progressive IC, is an interesting future work.

### Acknowledgements

ZZ and GG acknowledge partial support from DTRA grants HDTRA1-09-1-0050 and HDTRA1-14-1-0040, DoD ARO MURI Award W911NF-12-1-0385, and NSF grants CNS-10171647, CNS-1117536 and CNS-1411636. AB and AA acknowledge partial support from NSF grants IIS-1447566, IIS-1422557, CCF-1451986, CNS-1314560, IIS-0953274, IIS-1029711, NASA grant NNX12AQ39A, a gift from IBM and Yahoo!, and technical support from the University of Minnesota Supercomputing Institute.

## References

- [1] E. Even-Dar and A. Shapira, “A note on maximizing the spread of influence in social networks,” in *WINE*, 2007, pp. 281 – 286.
- [2] D. Kempe, J. Kleinberg, and v. Tardos, “Maximizing the spread of influence through a social network,” in *KDD*, 2003, pp. 137 – 146.
- [3] P. Domingos and M. Richardson, “Mining the net value of customers,” in *KDD*, 2001, pp. 57 – 66.
- [4] W. Chen, Y. Yuan, and L. Zhang, “Scalable influence maximization in social networks under the linear threshold model,” in *ICDM*, 2010, pp. 88 – 97.
- [5] W. Chen, C. Wang, and Y. Wang, “Scalable influence maximization for prevalent viral marketing in large-scale social networks,” in *KDD*, 2010, pp. 1029 – 1038.
- [6] M. Gomez-Rodriguez, D. Balduzzi, and B. Schlkopf, “Uncovering the temporal dynamics of diffusion networks,” in *ICML*, 2011.
- [7] N. Due, L. Song, M. G. Rodriguez, and H. Zha, “Scalable influence estimation in continuous-time diffusion networks,” in *NIPS*, 2013.
- [8] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance, “Cost-effective outbreak detection in networks,” in *KDD*, 2007, pp. 420 – 429.
- [9] A. Goyal, W. Lu, and L. V. Lakshmanan, “Simpath: An efficient algorithm for influence maximization under the linear threshold model,” in *ICDM*, 2011, pp. 211–220.
- [10] P. Clifford and A. Sudbury, “A model for spatial conflict,” *Biometrika*, vol. 60, no. 3, pp. 581 – 588, 1973.
- [11] R. Holley and T. Liggett, “Ergodic theorems for weakly interacting infinite systems and the voter model,” *The Annals of Probability*, vol. 3, no. 4, pp. 643 – 663, 1975.
- [12] M. Cha, A. Mislove, and K. P. Gummadi, “A measurement-driven analysis of information propagation in the flickr social network,” in *WWW*, 2009, pp. 721 – 730.
- [13] G. Lawler, *Random walk and the heat equation*, 2010.
- [14] P. G. Doyle and J. L. Snell, *Random walks and electric networks*, 1984.
- [15] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, “An analysis of approximations for maximizing submodular set functionsI,” *Mathematical Programming*, vol. 14, no. 1, pp. 265 – 294, Dec. 1978.
- [16] N. Pathak, A. Banerjee, and J. Srivastava, “A generalized linear threshold model for multiple cascades,” in *ICDM*, 2010, pp. 965 – 970.
- [17] F. P. Incropera, *Fundamentals of heat and mass transfer*. John Wiley & Sons, 2011.
- [18] D. Aldous and J. A. Fill, *Reversible Markov chains and random walks on graphs*, 2002.
- [19] P. Erdos and A. Renyi, “On the evolution of random graphs,” in *Pub. of the Mathematical Institute of the Hungarian Academy of Science*, 1960, pp. 17– 61.
- [20] A. Clauset, C. Moore, and M. E. J. Newman, “Hierarchical structure and the prediction of missing links in networks,” *Nature*, vol. 453, pp. 98–101, 2008.
- [21] J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos, and Z. Ghahramani, “Kronecker graphs: An approach to modeling networks,” *JMLR*, vol. 11, pp. 985–1042, 2010.
- [22] W. Zachary, “An information flow model for conflict and fission in small groups,” *Journal of Anthropological Research*, vol. 33, pp. 452 – 473, 1977.
- [23] L. A. Adamic and N. Glance, “The political blogosphere and the 2004 U.S. election: Divided they blog,” in *LinkKDD*, 2005, pp. 36 – 43.
- [24] J. Leskovec, D. Huttenlocher, and J. Kleinberg, “Predicting positive and negative links in online social networks,” in *WWW*, 2010, pp. 641 – 650.
- [25] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, “ArnetMiner: extraction and mining of academic social networks,” in *KDD*, 2008, pp. 990–998.
- [26] M. Kimura and K. Saito, “Tractable models for information diffusion in social networks,” in *PKDD*, 2006, pp. 259–271.
- [27] M. Gomez-Rodriguez and B. Schlkopf, “Influence maximization in continuous time diffusion networks,” in *ICML*, 2012.
- [28] S. Brin and L. Page, “The anatomy of a large-scale hypertextual web search engine,” 1998.
- [29] J. Tang, D. Zhang, and L. Yao, “Social network extraction of academic researchers,” in *ICDM*, 2007, pp. 292–301.
- [30] F. Zhang, *The Schur complement and its applications*. Springer, 2006, vol. 4.
- [31] A. Agarwal and J. Lang, *Foundations of analog & digital electronic circuits*, 2005.

---

# Scalable Recommendation with Hierarchical Poisson Factorization

---

**Prem Gopalan**

Department of Computer Science  
Princeton University  
Princeton, NJ

**Jake M. Hofman**

Microsoft Research  
641 Sixth Avenue, Floor 7  
New York, NJ

**David M. Blei**

Departments of Statistics & Computer Science  
Columbia University  
New York, NY

## Abstract

We develop hierarchical Poisson matrix factorization (HPF), a novel method for providing users with high quality recommendations based on implicit feedback, such as views, clicks, or purchases. In contrast to existing recommendation models, HPF has a number of desirable properties. First, we show that HPF more accurately captures the long-tailed user activity found in most consumption data by explicitly considering the fact that users have finite attention budgets. This leads to better estimates of users' latent preferences, and therefore superior recommendations, compared to competing methods. Second, HPF learns these latent factors by only explicitly considering positive examples, eliminating the often costly step of generating artificial negative examples when fitting to implicit data. Third, HPF is more than just one method—it is the simplest in a class of probabilistic models with these properties, and can easily be extended to include more complex structure and assumptions. We develop a variational algorithm for approximate posterior inference for HPF that scales up to large data sets, and we demonstrate its performance on a wide variety of real-world recommendation problems—users rating movies, listening to songs, reading scientific papers, and reading news articles.

## 1 INTRODUCTION

Recommendation systems are a vital component of the modern Web. They help readers effectively navigate otherwise unwieldy archives of information and help websites direct users to items—movies, articles, songs, products—that they will like. A recommendation system is built from historical data about which items each user has consumed, be it clicked, viewed, rated, or purchased. First, it uncovers

the behavioral patterns that characterize various types of users and the kinds of items they tend to like. Then, it exploits these discovered patterns to recommend future items to its users.

In this paper, we develop Hierarchical Poisson factorization (HPF) for generating high-quality recommendations. Our algorithms easily scale to massive data and outperform several existing methods. We show HPF is tailored to real-world properties of user behavior data: the heterogeneous interests of users, the varied types of items, and a realistic distribution of the finite resources that users have to consume these items.

In more detail, HPF is a probabilistic model of users and items. It associates each user with a latent vector of preferences, each item with a latent vector of attributes, and constrains both sets of vectors to be sparse and non-negative. The model assumes that each cell of the observed behavior matrix is drawn from a Poisson distribution—an exponential family distribution over non-negative integers—whose parameter is a linear combination of the corresponding user preferences and item attributes. The main computational problem is posterior inference: given an observed matrix of user behavior, we would like to discover the latent attributes that describe the items and the latent preferences of the users, which we can then use to make predictions and recommendations.

This inferential computation is common to many variants of matrix factorization. We find, however, that HPF enjoys significant quantitative advantages over classical methods for a variety of *implicit feedback* data sets. Figure 4 shows that HPF performs better than competing methods—including the industry standard of matrix factorization with user and item biases (MF) fit using stochastic gradient descent—for large data sets of Netflix users watching movies, Last.FM users listening to music, scientists reading papers, and *New York Times* readers clicking on articles.

We review related work in detail in Section 4. We now discuss details of the Poisson factorization model, including



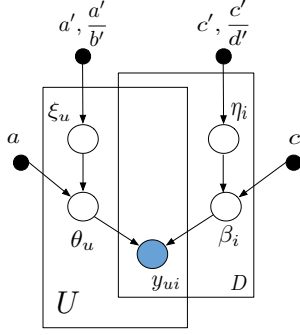


Figure 1: The hierarchical Poisson factorization model.

its statistical properties and methods for scalable inference.

## 2 POISSON RECOMMENDATION

In this section we describe the Poisson factorization model for recommendation, and discuss its statistical properties.

We are given data about users and items, where each user has consumed and possibly rated a set of items. The observation  $y_{ui}$  is the rating that user  $u$  gave to item  $i$ , or zero if no rating was given. In the “implicit” consumer data that we consider here,  $y_{ui}$  equals one if user  $u$  consumed item  $i$  and zero otherwise. User behavior data, such as purchases, clicks, or views, are typically sparse. Most of the values of the matrix  $y$  are zero.

We model these data with factorized Poisson distributions [4], where each item  $i$  is represented by a vector of  $K$  latent attributes  $\beta_i$  and each user  $u$  by a vector of  $K$  latent preferences  $\theta_u$ . The observations  $y_{ui}$  are modeled with a Poisson distribution, parameterized by the inner product of the user preferences and item attributes,  $y_{ui} \sim \text{Poisson}(\theta_u^\top \beta_i)$ . This is a variant of probabilistic matrix factorization [33] but where each user and item’s weights are positive [25] and where the Poisson replaces the Gaussian. While a Bernoulli distribution may seem more appropriate for modeling binary data, we demonstrate in Section 2.1 that the additivity of independent Poissons result in models that capture the marginal user, item distributions well.<sup>1</sup>

Beyond the basic data generating distribution, we place Gamma priors on the latent attributes and latent preferences, which encourage the model towards sparse representations of the users and items. Furthermore, we place additional priors on the user and item-specific rate parameter of those Gammas, which controls the average size of the representation. This hierarchical structure allows us to

<sup>1</sup>Our ongoing work considers censored Poisson distributions. Our initial results indicate that it is computationally expensive but does not give better performance.

capture the diversity of users, some tending to consume more than others, and the diversity of items, some being more popular than others. The literature on recommendation systems suggests that a good model must capture such heterogeneity across users and items [23].

Putting this together, the generative process of the hierarchical Poisson factorization model (HPF), illustrated in the graphical model in Figure 1, is as follows:

1. For each user  $u$ :
  - (a) Sample activity  $\xi_u \sim \text{Gamma}(a', a'/b')$ .
  - (b) For each component  $k$ , sample preference

$$\theta_{uk} \sim \text{Gamma}(a, \xi_u).$$

2. For each item  $i$ :
  - (a) Sample popularity  $\eta_i \sim \text{Gamma}(c', c'/d')$ .
  - (b) For each component  $k$ , sample attribute

$$\beta_{ik} \sim \text{Gamma}(c, \eta_i).$$

3. For each user  $u$  and item  $i$ , sample rating

$$y_{ui} \sim \text{Poisson}(\theta_u^\top \beta_i).$$

This process describes the statistical assumptions behind the model. We note that this contains, as a sub-class, a factorization model with fixed rate parameters for all users and items. We call this model Bayesian Poisson Factorization (BPF).

The central computational problem is posterior inference, which is akin to “reversing” the generative process. Given a user behavior matrix, we want to estimate the conditional distribution of the latent per-user and per-item structure,  $p(\theta_{1:N}, \beta_{1:M} | y)$ , termed the posterior, which is the key to recommendation. We estimate the posterior expectation of each user’s preferences, each items attributes and, subsequently, form predictions about which unconsumed items each user will like. We discuss posterior inference in Section 2.2.

Once the posterior is fit, we use HPF to recommend items to users by predicting which of the unconsumed items each will like. We rank each user’s unconsumed items by their posterior expected Poisson parameters,

$$\text{score}_{ui} = E[\theta_u^\top \beta_i | y]. \quad (1)$$

This amounts to asking the model to rank by probability which of the presently unconsumed items each user will likely consume in the future.

### 2.1 Properties of HPF

With the modeling details in place, we highlight several statistical properties of hierarchical Poisson factorization.

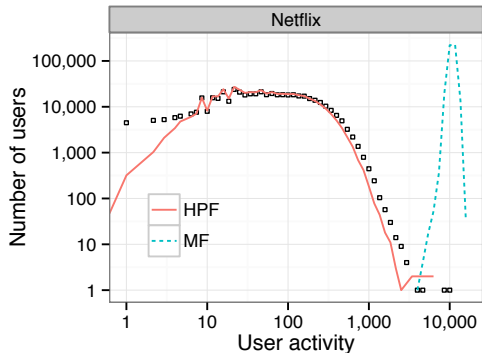


Figure 2: A posterior predictive check of the distribution of total ratings for the Netflix data set. The black squares show the empirical count of the number of users who have rated a given number of items, while the red and blue curves show the simulated totals from fitted Poisson and traditional matrix factorization models, respectively. The Poisson marginal closely matches the empirical, with the exception of users with very low activity, whereas classical matrix factorization fits a large mean to account for skew in the distribution and the missing ratings.

These properties provide advantages over *classical Gaussian matrix factorization*. Specifically, by classical MF we mean L2 regularized matrix factorization with bias terms for users and items, fit using stochastic gradient descent [23]. Without the bias terms, this corresponds to maximum a-posteriori inference under Probabilistic Matrix Factorization [33]. We generate negatives by randomly sampling from missing ratings in the training set [7, 8, 29].

**HPF captures sparse factors.** As mentioned above, the Gamma priors on preferences and attributes encourages sparse representations of users and items. Specifically, by setting the shape parameter to be small, most of the weights will be close to zero and only a few will be large. This leads to a simpler, more interpretable model.

**HPF models the long-tail of users and items.** One statistical characteristic of real-world user behavior data is the distribution of user activity (i.e., how many items a user consumed) and item popularity (i.e., how many users consumed an item). These distributions tend to be long-tailed: while most users consume a handful few items, a few “tail users” consume thousands of items. A question we can ask of a statistical model of user behavior data is how well it captures these distributions. We found that HPF captures them well, while classical matrix factorization does not.

To check this, we implemented a *posterior predictive check* (PPC) [31, 10], a technique for model assessment from the Bayesian statistics literature. The idea behind a PPC is to simulate a complete data set from the posterior predictive distribution—the distribution over data that the posterior

induces—and then compare the generated data set to the true observations. A good model will produce data that captures the important characteristics of the observed data.

We developed a PPC for matrix factorization algorithms on user behavior data. First, we formed posterior estimates of user preferences and item attributes for both classical MF and HPF. Then, from these estimates, we simulated user behavior by drawing values for each user and item. (For classical matrix factorization, we truncated these values at zero and rounded to one in order to generate a plausible matrix.) Finally, we compared the matrix generated by the posterior predictive distribution to the true observations.

Figure 2 illustrates our PPC for the Netflix data. In this figure, we illustrate three distributions over user activity: the observed distribution (squares), the distribution from a data set replicated by HPF (red line), and a distribution from a data set replicated by Gaussian MF with generated negatives using popularity-based sampling (blue line). HPF captures the truth much more closely than Gaussian MF, which overestimates the distribution of user activity. This indicates that HPF better represents real data when measured by its ability to capture distributions of user activity. In fact, this is encoded in its assumptions. We can rewrite the Poisson observation model as a two stage process where a user  $u$  first decides on a budget  $b_u$  she has to spend on items, and then spends this budget rating items that she is interested in:

$$b_u \sim \text{Poisson}(\theta_u^T \sum_i \beta_i)$$

$$[y_{u1}, \dots, y_{uM}] \sim \text{Mult}(b_u, \frac{\theta_u^T \beta_i}{\theta_u^T \sum_i \beta_i}).$$

This shows that learning a PF model for user-item ratings is effectively the same as learning a budget for each user while also learning how that budget is distributed across items.

**HPF downweights the effect of zeros.** Another advantage of HPF is that it implicitly down-weights the contribution of the items that each user did not consume. With an appropriate fit to user activity, the model has two ways of explaining an unconsumed item: either the user is not interested in it or she would be interested in if she the opportunity to consider it. In contrast, a user that consumes an item must be interested in it. Thus, the model benefits more from making latent factors for a consumed user/item pair more similar compared to making them less similar for an unconsumed user/item pair.

Classical MF is based on Gaussian likelihoods (i.e., squared loss), which gives equal weight to consumed and unconsumed items. Consequently, when faced with a sparse matrix and implicit feedback, i.e., binary consumption data, matrix factorization places more total emphasis on the unconsumed user/item pairs. (This too can be seen

to stem from classical MF’s overestimation of the distribution of user activity.) To address this, researchers have patched MF in complex ways, for example, by including per-observation confidences [23] or considering all zeroes to be hidden variables [29]. Poisson factorization naturally solves this problem with a realistic model of user activity.

As an example, consider two similar science fiction movies, “Star Wars” and “The Empire Strikes Back”, and consider a user who has seen one of them. The Gaussian model pays an equal penalty for making the user similar to these items as it does for making the user different from them—with quadratic loss, seeing “Star Wars” is evidence for liking science fiction, but not seeing “The Empire Strikes Back” is evidence for disliking it. The Poisson model, however, will prefer to bring the user’s latent weights closer to the movies’ weights because it favors the information from the user watching “Star Wars”. Further, because the movies are similar, this increases the Poisson model’s predictive score that a user who watches “Star Wars” will also watch “The Empire Strikes Back”.

**Fast inference with sparse matrices.** Finally, the likelihood of the observed data under HPF depends only on the consumed items, that is, the non-zero elements of the user/item matrix  $y$ . This facilitates computation for the kind of sparse matrices we observe in real-world data.

We can see this property from the form of the Poisson distribution. Given the latent preferences  $\theta_u$  and latent attributes  $\beta_i$ , the Poisson distribution of the rating  $y_{ui}$  is

$$p(y_{ui} | \theta_u, \beta_i) = (\theta_u^\top \beta_i)^{y_{ui}} \exp\{-\theta_u^\top \beta_i\} / y_{ui}! \quad (2)$$

Recall the elementary fact that  $0! = 1$ . With this, the log probability of the complete matrix  $y$  can be written as

$$\log p(y | \theta, \beta) = \sum_{\{y_{ui} > 0\}} y_{ui} \log(\theta_u^\top \beta_i) - \log y_{ui}! - (\sum_u \theta_u)^\top (\sum_i \beta_i).$$

This avoids the need for sub-sampling [7], approximation [17], or stochastic optimization [27] that complicate other approaches.

## 2.2 INFERENCE WITH VARIATIONAL METHODS

Using HPF for recommendation hinges on solving the posterior inference problem. Given a set of observed ratings, we would like to infer the user preferences and item attributes that explain these ratings, and then use these inferences to recommend new content to the users. In this section we discuss the details and practical challenges of posterior inference for HPF, and present a mean-field variational inference algorithm as a scalable approach. Our algorithm easily accommodates data sets with millions of users and hundreds of thousands of items on a single CPU.

Given a matrix of user behavior, we would like to compute the posterior distribution of user preferences  $\theta_{uk}$ , item attributes  $\beta_{ik}$ , user activity  $\xi_u$  and item popularity  $\eta_i$ . As for many Bayesian models, however, the exact posterior is computationally intractable. We show how to efficiently approximate the posterior with mean-field variational inference.

Variational inference is an optimization-based strategy for approximating posterior distributions in complex probabilistic models [21, 35]. Variational algorithms posit a family of distributions over the hidden variables, indexed by free “variational” parameters, and then find the member of that family that is closest in Kullback-Liebler (KL) divergence to the true posterior. (The form of the family is chosen to make this optimization possible.) Thus, variational inference turns the inference problem into an optimization problem. Variational inference has previously been used for large-scale recommendation [29].

We will describe a simple variational inference algorithm for HPF. To do so, however, we first give an alternative formulation of the model in which we add an additional layer of latent variables. These auxiliary variables facilitate derivation and description of the algorithm [11, 16].

For each user and item we add  $K$  latent variables  $z_{uik} \sim \text{Poisson}(\theta_{uk}\beta_{ik})$ , which are integers that sum to the user/item value  $y_{ui}$ . A sum of Poisson random variables is itself a Poisson with rate equal to the sum of the rates. Thus, these new latent variables preserve the marginal distribution of the observation,  $y_{ui} \sim \text{Poisson}(\theta_u^\top \beta_i)$ . These variables can be thought of as the contribution from component  $k$  to the total observation  $y_{ui}$ . Note that when  $y_{ui} = 0$ , these auxiliary variables are not random—the posterior distribution of  $z_{uik}$  will place all its mass on the zero vector. Consequently, our inference procedure need only consider  $z_{uik}$  for those user/item pairs where  $y_{ui} > 0$ .

With these latent variables in place, we now describe the algorithm. First, we posit the variational family over the hidden variables. Then we show how to optimize its parameters to find an approximation to the posterior.

The latent variables in the model are user weights  $\theta_{uk}$ , item weights  $\beta_{ik}$ , and user-item contributions  $z_{uik}$ , which we represent as a  $K$ -vector of counts  $z_{ui}$ . The *mean-field family* considers these variables to be independent and each governed by its own distribution,

$$q(\beta, \theta, \xi, \eta, z) = \prod_{i,k} q(\beta_{ik} | \lambda_{ik}) \prod_{u,k} q(\theta_{uk} | \gamma_{uk}) \prod_u q(\xi_u | \kappa_u) \prod_i q(\eta_i | \tau_i) \prod_{u,i} q(z_{ui} | \phi_{ui}).$$

Though the variables are independent, this is a flexible family of distributions because each variable is governed by its own free parameter. The variational factors for preferences  $\theta_{uk}$ , attributes  $\beta_{ik}$ , activity  $\xi_u$ , and popularity  $\eta_i$  are

For all users and items, initialize the user parameters  $\gamma_u, \kappa_u^{\text{rte}}$  and item parameters  $\lambda_i, \tau_i^{\text{rte}}$  to the prior with a small random offset. Set the user activity and item popularity shape parameters:

$$\kappa_u^{\text{shp}} = a' + Ka; \quad \tau_i^{\text{shp}} = c' + Kc$$

Repeat until convergence:

1. For each user/item such that  $y_{ui} > 0$ , update the multinomial:

$$\phi_{ui} \propto \exp\{\Psi(\gamma_{uk}^{\text{shp}}) - \log \gamma_{uk}^{\text{rte}} + \Psi(\lambda_{ik}^{\text{shp}}) - \log \lambda_{ik}^{\text{rte}}\}.$$

2. For each user, update the user weight and activity parameters:

$$\begin{aligned} \gamma_{uk}^{\text{shp}} &= a + \sum_i y_{ui} \phi_{uik} \\ \gamma_{uk}^{\text{rte}} &= \frac{\kappa_u^{\text{shp}}}{\kappa_u^{\text{rte}}} + \sum_i \lambda_{ik}^{\text{shp}} / \lambda_{ik}^{\text{rte}} \\ \kappa_u^{\text{rte}} &= \frac{a'}{b'} + \sum_k \frac{\gamma_{uk}^{\text{shp}}}{\gamma_{uk}^{\text{rte}}} \end{aligned}$$

3. For each item, update the item weight and popularity parameters:

$$\begin{aligned} \lambda_{ik}^{\text{shp}} &= c + \sum_u y_{ui} \phi_{uik} \\ \lambda_{ik}^{\text{rte}} &= \frac{\tau_i^{\text{shp}}}{\tau_i^{\text{rte}}} + \sum_u \gamma_{uk}^{\text{shp}} / \gamma_{uk}^{\text{rte}} \\ \tau_i^{\text{rte}} &= \frac{c'}{d'} + \sum_k \frac{\lambda_{ik}^{\text{shp}}}{\lambda_{ik}^{\text{rte}}} \end{aligned}$$

Figure 3: Variational inference for Poisson factorization. Each iteration only needs to consider the non-zero elements of the user/item matrix.

all Gamma distributions, with freely set scale and rate variational parameters. The variational factor for  $z_{ui}$  is a free multinomial, i.e.,  $\phi_{ui}$  is a  $K$ -vector that sums to one. This form stems from  $z_{ui}$  being a bank of Poisson variables conditional on a fixed sum  $y_{ui}$ , and the property that such conditional Poissons are distributed as a multinomial [20, 5].

After specifying the family, we fit the variational parameters  $\nu = \{\lambda, \gamma, \kappa, \tau, \phi\}$  to minimize the KL divergence to the posterior, and then use the corresponding variational distribution  $q(\cdot | \nu^*)$  as its proxy. The mean-field factorization facilitates both optimizing the variational objective and downstream computations with the approximate posterior, such as the recommendation score of Equation 1.

We optimize the variational parameters with a coordinate ascent algorithm, iteratively optimizing each parameter while holding the others fixed. The algorithm is illustrated

in Figure 3. We denote shape with the superscript “shp” and rate with the superscript “rte”. We provide a detailed derivation in the Appendix.

Note that our algorithm is efficient on sparse matrices. In step 1, we need only update variational multinomials for the non-zero user/item observations  $y_{ui}$ . In steps 2 and 3, the sums over users and items need only to consider non-zero observations. This efficiency is thanks the likelihood of the full matrix only depending on the non-zero observations, as we discussed in the previous section. Both HPF and BPF enjoy this property and have the same computational overhead, but HPF allows for more flexibility in modeling the variation in activity and popularity across users and items, respectively.

We terminate the algorithm when the variational distribution converges. Convergence is measured by computing the prediction accuracy on a validation set. Specifically, we approximate the probability that a user consumed an item using the variational approximations to posterior expectations of  $\theta_u$  and  $\beta_i$ , and compute the average predictive log likelihood of the validation ratings. The HPF algorithm stops when the change in log likelihood is less than 0.0001%. We find that the algorithm is largely insensitive to small changes in the hyper-parameters. To enforce sparsity, we set the shape hyperparameters  $a', a, c$  and  $c'$  to provide exponentially shaped prior Gamma distributions—we fixed each hyperparameter at 0.3. We set the hyperparameters  $b'$  and  $d'$  to 1, fixing the prior mean at 1.

### 3 EMPIRICAL STUDY

We evaluate the performance of the Hierarchical Poisson factorization (HPF) algorithm on a variety of large-scale user behavior data sets: users listening to music, users watching movies, users reading scientific articles, and users reading the newspaper. We find that HPF provides significantly better recommendations than competing methods. We provide an exploratory analysis of preferences and attributes on the New York Times data set in the appendix.<sup>2</sup>

**Data Sets.** We study the HPF algorithm in Figure 3 on several data sets of user behavior:

- The **Mendeley** data set [19] of scientific articles is a binary matrix of 80,000 users and 260,000 articles, with 5 million observations. Each cell indicates the presence or absence of an article in a user’s library.
- The **Echo Nest** music data set [2] is a matrix of 1 million users and 385,000 songs, with 48 million observations. Each observation is the number of times a user played a song.

<sup>2</sup>Our source code is available from <https://github.com/premgopalan/hgaprec>

- The **New York Times** data set is a matrix of 1,615,675 users and 103,390 articles, with 80 million observations. Each observation is the number of times a user viewed an article.
- The **Netflix** data set [23] contains 480,000 users and 17,770 movies, with 100 million observations. Each observation is the rating (from 1 to 5 stars) that a user provided for a movie.

The scale and diversity of these data sets enables a robust evaluation of our algorithm. The Mendeley, Echo Nest, and New York Times data are sparse compared to Netflix. For example, we observe only 0.001% of all possible user-item ratings in Mendeley, while 1% of the ratings are non-zero in the Netflix data. This is partially a reflection of large number of items relative to the number of users in these data sets.

Furthermore, the intent signaled by an observed rating varies significantly across these data sets. For instance, the Netflix data set gives the most direct measure of stated preferences for items, as users provide a star rating for movies they have watched. In contrast, article click counts in the New York Times data are a less clear measure of how much a user likes a given article—most articles are read only once, and a click through is only a weak indicator of whether the article was fully read, let alone liked. Ratings in the Echo Nest data presumably fall somewhere in between, as the number of times a user listens to a song likely reveals some indirect information about their preferences.

As such, we treat each data set as a source of implicit feedback, where an observed positive rating indicates that a user likes a particular item, but the rating value itself is ignored. The Mendeley data are already of this simple binary form. For the Echo Nest and New York Times data, we consider any song play or article click as a positive rating, regardless of the play or click count. As in previous work, we consider an implicit version of the Netflix data where only 4 and 5 star ratings are retained as observations [29].

**Competing methods.** We compare Poisson factorization against an array of competing methods:

- **NMF:** Non-negative Matrix Factorization [25]. In NMF, user preferences and item attributes are modeled as non-negative vectors in a low-dimensional space. These latent vectors are randomly initialized and modified via an alternating multiplicative update rule to minimize the Kullback-Leibler divergence between the actual and modeled rating matrices. We use the GraphLab implementation of NMF [24] to scale to our large data sets.
- **LDA:** Latent Dirichlet Allocation [3]. LDA is a Bayesian probabilistic generative model where user

preferences are represented by a distribution over different topics, and each topic is represented by a distribution over items. Interest and topic distributions are randomly initialized and updated using stochastic variational inference [16] to approximate these intractable posteriors. We used the default setting of the hyperparameters in the Vowpal Wabbit package [37].

- **MF:** Probabilistic Matrix Factorization with user and item biases. We use a variant of matrix factorization popularized through the Netflix Prize [23], where a linear predictor—comprised of a constant term, user activity and item popularity biases, and a low-rank interaction term—is fit to minimize the mean squared error between the predicted and observed rating values, subject to L2 regularization to avoid overfitting. Weights are randomly initialized and updated via stochastic gradient descent using the Vowpal Wabbit package [37]. This corresponds to maximum a-posteriori inference under Probabilistic Matrix Factorization [33]. We selected hyperparameters using grid search with a small validation set.
- **ClIMF:** Collaborative Less-is-More filtering [34] maximizes mean reciprocal rank to improve the top- $n$  predictive performance on binary relevance data sets. We use the GraphLab implementation of ClIMF [24] to scale to our large data sets, and use the default parameter settings in the package.

We note that while HPF and LDA take only the non-zero observed ratings as input, traditional matrix factorization requires that we provide explicit zeros in the ratings matrix as negative examples for the implicit feedback setting. In practice, this amounts to either treating all missing ratings as zeros (as in NMF) and down-weighting to balance the relative importance of observed and missing ratings [17], or generating negatives by randomly sampling from missing ratings in the training set [8, 7, 29]. We take the latter approach for computational convenience, employing a popularity-based sampling scheme: we sample users by activity—the number of items rated in the training set—and items by popularity—the number of training ratings an item received to generate negative examples.<sup>3</sup>

Finally, we note that a few candidate algorithms failed to scale to our data sets. The fully Bayesian treatment of the Probabilistic Matrix Factorization [32], uses a MCMC algorithm for inference. The authors [32] report that a single Gibbs iteration on the Netflix data set with 60 latent factors, requires 30 minutes, and that they throw away the first 800 samples. This implies at least 16 days of training, while the HPF variational inference algorithm converges within

<sup>3</sup>We also compared this to a uniform random sampling of negative examples, but found that the popularity-based sampling performed better.



Figure 4: Predictive performance on data sets. The top and bottom plots show normalized mean precision and mean recall at 20 recommendations, respectively. While the relative performance of the competing methods varies across data sets, HPF consistently outperforms each of them.

13 hours on the Netflix data. Another alternative, Bayesian Personalized Ranking (BPR) [30, 8], optimizes a ranking-based criteria using stochastic gradient descent. The algorithm performs an expensive bootstrap sampling step at each iteration to generate negative examples from the vast set of unobserved. We found time and space constraints to be prohibitive when attempting to use BPR with the data sets considered here. Finally, the GraphChi implementation of CLiMF [24] failed with an error on the Netflix and New York Times data sets.

**Evaluation.** Prior to training any models, we randomly select 20% of ratings in each data set to be used as a held-out test set comprised of items that the user has consumed. Additionally, we set aside 1% of the training ratings as a validation set and use it to determine algorithm convergence and to tune free parameters. We used the HPF settings described in Section 2.2 across all data sets, and set the number of latent components  $K$  to 100.

During testing, we generate the top  $M$  recommendations for each user as those items with the highest predictive score under each method. For each user, we compute a variant of precision-at- $M$  that measures the fraction of relevant items in the user’s top- $M$  recommendations. So as not to artificially deflate this measurement for lightly active users who have consumed fewer than  $M$  items, we compute *normalized* precision-at- $M$ , which adjusts the denominator to be at most the number of items the user has in the test set. Likewise, we compute recall-at- $M$ , which captures the fraction of items in the test set present in the top  $M$  recommendations.

Figure 4 shows the normalized mean precision at 20 recommendations for each method and data sets. We see that

HPF outperforms other methods on all data sets by a sizeable margin. Poisson factorization provides high-quality recommendations—a relatively high fraction of items recommended by HPF are found to be relevant, and many relevant items are recommended. While not shown in these plots, the relative performance of methods within a data set is consistent as we vary the number of recommendations shown to users. We also note that while Poisson factorization dominates across all of these data sets, the relative quality of recommendations from competing methods varies substantially from one data set to the next. For instance, LDA performs quite well on the Echo Nest data, but fails to beat classical matrix factorization for the implicit Netflix data set.

We also study precision and recall as a function of user activity to investigate how performance varies across users of different types. In particular, Figure 5 shows the mean difference in precision and recall to HPF, at 20 recommendations, as we look at performance for users of varying activity, measured by percentile. For example, the 10% mark on the x-axis shows mean performance across the bottom 10% of users, who are least active; the 90% mark shows the mean performance for all but the top 10% of most active users. Here we see that Poisson factorization outperforms other methods for users of all activity levels.

## 4 RELATED WORK

The roots of Poisson factorization come from nonnegative matrix factorization [25], where the objective function is equivalent to a factorized Poisson likelihood. The original NMF update equations have been shown to be an expectation-maximization (EM) algorithm for maximum

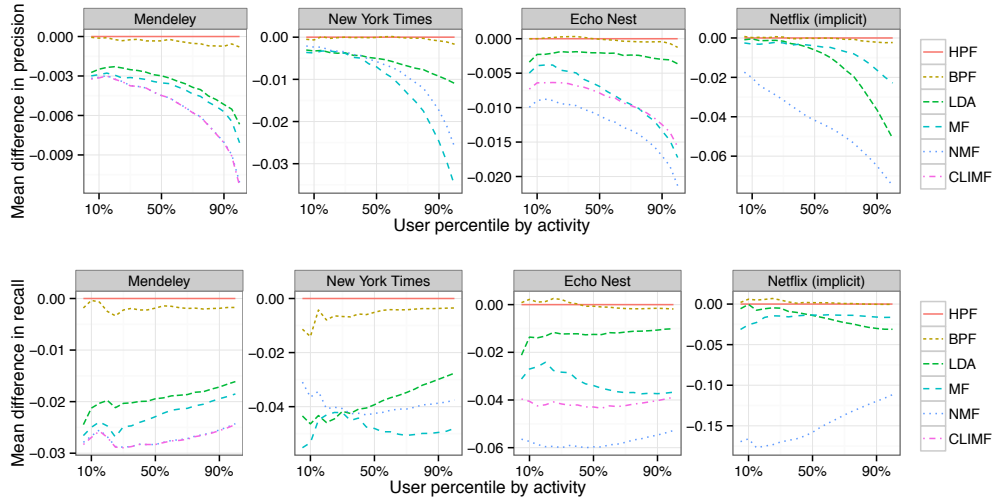


Figure 5: Predictive performance across users. The top and bottom plots show the mean difference in precision and recall to HPF at 20 recommendations, respectively, by user activity.

likelihood estimation of a Poisson model [5].

Placing a Gamma prior on the user weights results in the GaP model [4], which was developed as an alternative text model to latent Dirichlet allocation (LDA) [3, 18]. The GaP model is fit using the expectation-maximization algorithm to obtain point estimates for user preferences and item attributes. The Probabilistic Factor Model (PFM) [26] improves upon GaP by placing a Gamma prior on the item weights as well, and using multiplicative update rules to infer an approximate maximum a posteriori estimate of the latent factors. Our model uses a hierarchical prior structure of Gamma priors on both user and item weights, and Gamma priors over the rate parameters from which these weights are drawn. Furthermore, we approximate the full posterior over all latent factors using a scalable variational inference algorithm.

Independently of GaP and user behavior models, Poisson factorization has been studied in the context of signal processing for source separation [5, 15] and for detecting community structure in network data [1, 13]. This research includes variational approximations to the posterior, though the issues and details around these data differ significantly from user data we consider and our derivation in the supplement (based on auxiliary variables) is more direct.

When modeling implicit feedback data sets, researchers have proposed merging factorization techniques with neighborhood models [22], weighting techniques to adjust the relative importance of positive examples [17], and sampling-based approaches to create informative negative examples [8, 7, 29]. In addition to the difficulty in appropriately weighting or sampling negative examples, there is a known selection bias in provided ratings that causes further complications [28]. HPF does not require such special

adjustments for negative examples and scales linearly in the observed ratings.

**Comparison to Gaussian MF.** Many of the leading MF methods are based on Gaussian likelihoods (i.e., squared loss). When applied to explicit data, Gaussian models are fit only to the observed ratings [23] and infer distributions over user preferences. For each user, the items she did not consume, i.e., the zero observations, are treated as missing. Gaussian models make up the state of the art in this setting [32, 33, 23].

In implicit data sets of user consumption, there is a fundamental asymmetry that allows one to infer which items a user consumed, and therefore liked, but not which items a user did not like [17]. In this setting, Gaussian MF applied to all observations gives equal weight to consumed and unconsumed items. Consequently, when faced with a sparse matrix and implicit feedback, matrix factorization places more total emphasis on the unconsumed user/item pairs.

To address this limitation of Gaussian MF, researchers have proposed two main approaches. The first approach, proposed by [17], is to treat the unconsumed items with greater uncertainty and increase confidence as the rating for an item increases. This converts the raw observations into two separate quantities with distinct interpretations: user preferences and confidence levels. Hu et al. [17] present an alternating least squares algorithm that considers all observations but whose per-iteration complexity is still linear in the number of non-zero observations.

The second approach is to randomly synthesize negative examples [7, 8, 29]. In this approach, unconsumed items are subsampled for each user to balance out the consumed items. As Dror et al. [7] note, it is unclear how to bal-

ance these two sets of items. Do we use an equal number of consumed and unconsumed items, or do we use the full set of unconsumed items [6, 17]? Further, the subsampling of negative or unconsumed items is often expensive, and can account for a substantial fraction of resources devoted to model fitting. An issue that we found in Gaussian MF with subsampled zeros, fit using SGD, is that it systematically overestimates the users' budgets. We confirmed this in Section 3 using a posterior predictive check [10]. Poisson factorization does not require synthesizing negative examples and is better able to capture distributions of users' budgets.

Further, the HPF algorithm retains the linear-scaling of Gaussian MF with downweighted zeros [17]. HPF algorithms only need to iterate over the consumed items in the observed matrix of user behavior. This follows from the mathematical form of the Poisson distribution. In contrast, the subsampling-based Gaussian MF methods [7, 8, 29] must iterate over both positive and negative examples in the implicit setting. This makes it difficult to take advantage of data sparsity to scale to massive data sets.

Finally, unlike Gaussian MF which typically provides dense latent representations of users and items, PF models provide sparse latent representations. This property arises from the PF log-likelihood which can be shown to minimize the information (Kullback-Leibler) divergence under NMF [5], and from the Gamma priors in the HPF model.

**Recent extensions.** Building on the HPF model and algorithm we presented in a preprint, recent extensions have been proposed. One extension is a combined model of article text and reader preferences [14]. This model takes advantage of the sparse, non-negative representations in PF, which are useful in capturing different types of discrete data, such as word counts and user ratings. Further, they exploit the additive properties of independent Poisson random variables to capture dependencies between discrete data, for example, the dependence of user ratings of an article on its content. Another recent work proposes a Bayesian nonparametric model [12] that adapts the dimensionality of the latent representations, learning the preference patterns (and their number) that best describe the users. Both models exploit the scalability of PF algorithms to study massive data sets. These extensions testify to the modeling flexibility of PF models.

## 5 DISCUSSION

We have demonstrated that Poisson factorization is an efficient and effective means of generating high quality recommendations across a variety of data sets ranging from movie views to scientific article libraries. It significantly outperforms a number of leading methods in modeling implicit behavior data without the need for ad hoc modifications. Variational inference for HPF scales to massive data

and differs from traditional methods in its ability to capture the heterogeneity amongst users and items, accounting for the wide range of activity and popularity amongst them, respectively. The HPF algorithm is a robust, off-the-shelf tool, providing high accuracy even with fixed hyperparameter settings.

Finally, we emphasize that HPF is more than just one method—it is the simplest in a class of probabilistic models with these properties, and has already been extended to a combined model of article content and reader ratings [14], and a Bayesian nonparametric model that adapts the dimensionality of the latent representations [12].

A notable innovation in Gaussian MF is the algorithm of [17] that explicitly downweights zeros using confidence parameters. We presented the empirical study in this paper comparing to the Gaussian MF with subsampled zeros [23]. We attempted to compare to a GraphChi implementation [24] of [17], but it gave unexpectedly poor results. We found another implementation [36], and these comparisons are ongoing work. Another piece of ongoing work includes bringing the confidence-weighting of [17] into HPF. This will allow downweighting of the zeros beyond that provided implicitly by Poisson factorization.

## Acknowledgements

We thank Ulrich Paquet, Laurent Charlin, Francisco J. R. Ruiz, Rajesh Ranganath, Matthew D. Hoffman and the anonymous reviewers for comments and discussions. DMB is supported by NSF CAREER NSF IIS-0745520, NSF BIGDATA NSF IIS-1247664, ONR N00014-11-1-0651, and DARPA FA8750-14-2-0009.

## References

- [1] B. Ball, B. Karrer, and M. Newman. Efficient and principled method for detecting communities in networks. *Physical Review E*, 84(3), Sept. 2011.
- [2] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere. The million song dataset. In *ISMIR*, 2011.
- [3] D. Blei, A. Ng, and M. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, January 2003.
- [4] J. Canny. GaP: A factor model for discrete data. In *ACM SIGIR*, 2004.
- [5] A. T. Cemgil. Bayesian inference for nonnegative matrix factorisation models. *Computational Intelligence and Neuroscience*, 2009, May 2009.
- [6] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 39–46. ACM, 2010.
- [7] G. Dror, N. Koenigstein, Y. Koren, and M. Weimer. The Yahoo! music dataset and KDD-cup '11. *Journal of Machine Learning Research*, 18:8–18, 2012.



- [8] Z. Gantner, L. Drumond, C. Freudenthaler, and L. Schmidt-Thieme. Bayesian personalized ranking for non-uniformly sampled items. *JMLR W&CP*, Jan 2012.
- [9] A. Gelman, J. Carlin, H. Stern, and D. Rubin. *Bayesian Data Analysis*. Chapman & Hall, London, 1995.
- [10] A. Gelman, X. Meng, and H. Stern. Posterior predictive assessment of model fitness via realized discrepancies. *Statistica Sinica*, 6:733–807, 1996.
- [11] Z. Ghahramani and M. Beal. Propagation algorithms for variational Bayesian learning. In *NIPS*, pages 507–513, 2001.
- [12] P. Gopalan, F. J. Ruiz, R. Ranganath, and D. M. Blei. Bayesian nonparametric Poisson factorization for recommendation systems. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, pages 275–283, 2014.
- [13] P. K. Gopalan and D. M. Blei. Efficient discovery of overlapping communities in massive networks. *Proceedings of the National Academy of Sciences*, 110(36):14534–14539, 2013.
- [14] P. K. Gopalan, L. Charlin, and D. Blei. Content-based recommendations with Poisson factorization. In *Advances in Neural Information Processing Systems*, pages 3176–3184, 2014.
- [15] M. Hoffman. Poisson-uniform nonnegative matrix factorization. In *ICASSP*, 2012.
- [16] M. Hoffman, D. Blei, C. Wang, and J. Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14(1303–1347), 2013.
- [17] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM’08. Eighth IEEE International Conference on*, pages 263–272. IEEE, 2008.
- [18] D. Inouye, P. Ravikumar, and I. Dhillon. Admixture of Poisson MRFs: A topic model with word dependencies. In *ICML*, pages 683–691, 2014.
- [19] J. Jack, Kris anwd Hammerton, D. Harvey, J. J. Hoyt, J. Reichelt, and V. Henning. Mendeleys reply to the datatel challenge. *Procedia Computer Science*, 1(2):1–3, 2010.
- [20] N. Johnson, A. Kemp, and S. Kotz. *Univariate Discrete Distributions*. John Wiley & Sons, 2005.
- [21] M. Jordan, Z. Ghahramani, T. Jaakkola, and L. Saul. Introduction to variational methods for graphical models. *Machine Learning*, 37:183–233, 1999.
- [22] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *ACM SIGKDD*, pages 426–434. ACM, 2008.
- [23] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.
- [24] A. Kyrola, G. E. Blelloch, and C. Guestrin. Graphchi: Large-scale graph computation on just a PC. In *OSDI*, volume 12, pages 31–46, 2012.
- [25] D. Lee and H. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, October 1999.
- [26] H. Ma, C. Liu, I. King, and M. R. Lyu. Probabilistic factor models for web site recommendation. In *ACM SIGIR*, pages 265–274. ACM Press, 2011.
- [27] J. Mairal, J. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, 11:19–60, 2010.
- [28] B. Marlin, R. S. Zemel, S. Roweis, and M. Slaney. Collaborative filtering and the missing at random assumption. *arXiv preprint arXiv:1206.5267*, 2012.
- [29] U. Paquet and N. Koenigstein. One-class collaborative filtering with random graphs. In *WWW*, 2013.
- [30] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In *UAI*, pages 452–461, 2009.
- [31] D. Rubin. Bayesianly justifiable and relevant frequency calculations for the applied statistician. *The Annals of Statistics*, 12(4):1151–1172, 1984.
- [32] R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *ICML*, pages 880–887. ACM, 2008.
- [33] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. *Advances in Neural Information Processing Systems*, 20:1257–1264, 2008.
- [34] Y. Shi, A. Karatzoglou, L. Baltrunas, M. Larson, N. Oliver, and A. Hanjalic. Climf: learning to maximize reciprocal rank with collaborative less-is-more filtering. In *Proceedings of the sixth ACM conference on Recommender systems*, pages 139–146. ACM, 2012.
- [35] M. Wainwright and M. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1–2):1–305, 2008.
- [36] C. Wang and D. M. Blei. Collaborative topic modeling for recommending scientific articles. In *ACM SIGKDD, KDD ’11*, pages 448–456, 2011.
- [37] K. Weinberger, A. Dasgupta, J. Langford, A. Smola, and J. Attenberg. Feature hashing for large scale multitask learning. In *ICML*, pages 1113–1120, 2009.

---

# State Sequence Analysis in Hidden Markov Models

---

**Yuri Grinberg**

Ottawa Hospital Research Institute  
Ottawa, Ontario, Canada  
ygrinberg@ohri.ca  
www.perkinslab.ca

**Theodore J. Perkins**

Ottawa Hospital Research Institute  
Ottawa, Ontario, Canada  
tperkins@ohri.ca  
www.perkinslab.ca

## Abstract

Given a discrete time finite state hidden Markov model (HMM) and a sequence of observations, there are different ways to estimate the hidden behavior of the system. In this paper, the problem of finding the most probable state sequence is considered. The state sequence, as opposed to the state trajectory, specifies the sequence of states that the HMM visits but does not specify the dwelling times in these states. This inference problem is relevant in a variety of domains, like text analysis, speech recognition, or behavior recognition, where the exact timing of hidden state transitions is not nearly as important as the sequence of states visited. No existing algorithm addresses this inference question adequately. Leveraging previous work on continuous time Markov chains, we develop a provably correct algorithm, called *state sequence analysis*, that addresses this inference question in HMMs. We discuss and illustrate empirically the differences between finding the most probable state sequence directly and doing so through running the Viterbi algorithm and collapsing repetitive state visitations. Experimental results in two synthetic domains demonstrate that the Viterbi-based approach can be significantly suboptimal compared to state sequence analysis. Further, we demonstrate the benefits of the proposed approach on a real activity recognition problem.

## 1 INTRODUCTION

Hidden Markov models are a powerful and widely-used formalism for analyzing sequential information in a variety of domains, including speech recognition [Bahl et al., 1986, Rabiner, 1989], text analysis [Blei and Moreno, 2001], behavior recognition [Yamato et al., 1992, Nguyen et al., 2005], protein struc-

ture prediction [Sonnhammer et al., 1998], genomics [Haussler and Eeckman, 1996, Wang et al., 2007], and so on. As with graphical models generally, much of the utility of HMMs derives from our ability to use them to make estimates about hidden/unobserved variables based on observable variables. HMMs are often used as models of dynamical systems. In this context, the unobserved variables would be the true (underlying) state of the dynamical system. The observed variables would be something that we “see” when we observe or measure the system. The observations typically have some relationship to the underlying state, but the relationship may be imperfect or noisy. A typical task would then involve receiving some kind of observations of the system, and estimating the underlying states that generated those observations. Of course, an HMM can model sequential information that is not dynamical in nature, as seen for instance in applications in text analysis, genome annotation, etc. Regardless, the most common task is to estimate underlying states based on observations.

There are several different types of hidden state estimation problems for HMMs. Although we define HMMs formally in the next section, to distinguish different types of inference problems it is useful to introduce a small amount of notation here. We let  $X_t$  be a random variable denoting the underlying state at time  $t$ , and  $x_t$  a realization of that variable. Similarly, we let  $o_t$  be the observation at time  $t$ . Further, we let  $X_{1:t} = (X_1, X_2, \dots, X_t)$  be a random variable describing possible system *trajectories*, a realization of which is denoted  $\mathbf{x}_{1:t} = (x_1, x_2, \dots, x_t)$ . Similarly, a series of observations is denoted  $\mathbf{o}_{1:t} = (o_1, o_2, \dots, o_t)$ .

One of the fundamental HMM inference problems is to compute the probabilities of different underlying system states based on observations. More formally, if we receive a stream of observations,  $o_1, o_2, o_3, \dots$  then at each time  $t = 1, 2, 3, \dots$  we may want to compute the probabilities of different states  $x$  conditional on the observations:  $P(X_t = x | \mathbf{o}_{1:t})$ . This can be done by the forward algorithm [Rabiner, 1989], which is very efficient, and indeed allows us to readily compute  $P(X_t = x | \mathbf{o}_{1:t})$  incremen-

tally based on  $P(X_{t-1} = x' | \mathbf{o}_{1:t-1})$ . If we are given an entire sequence of observations,  $\mathbf{o}_{1:t}$ , and we want to estimate the underlying state probabilities at all times, namely  $P(X_{t'} = x | \mathbf{o}_{1:t})$  for all  $x$  and  $1 \leq t' \leq t$ , then the forward-backward algorithm [Rabiner, 1989] gives us an efficient solution. From those probabilities, one can easily compute the most probable state at each time:  $x_{t'}^{\max} = \arg \max_x P(X_{t'} = x | \mathbf{o}_{1:t})$ .

Each of these problems give us some information about underlying states based on observations, but they do not explicitly give us any pathway information. The forward and forward-backward algorithms, for example, give us only state probabilities. The sequence of most probable states,  $x_1^{\max}, x_2^{\max}, x_3^{\max}, \dots$ , may or may not comprise a valid system path. That is, some transition  $x_{t'}^{\max} \rightarrow x_{t'+1}^{\max}$  may not even be allowed under the dynamics. Thus, while these approaches are useful for estimating underlying states or “classifying” time points into different states, they are not directly useful for reconstructing underlying system paths.

For path reconstruction, by far the most common approach is the Viterbi algorithm [Rabiner, 1989], which is an efficient means for computing the maximum probability trajectory underlying a given sequence of observations:  $\arg \max_{\mathbf{x}_{1:t}} P(\mathbf{x}_{1:t} | \mathbf{o}_{1:t})$ . Despite the many successes of the Viterbi approach (e.g., in speech recognition, activity recognition and bioinformatics, as described above), it has been critiqued on a few different grounds. For one, like any *maximum a posteriori* (MAP) estimator, there is a question of how “representative” the maximum is. Intuitively, if the bulk of the posterior distribution contains trajectories that are “unlike” the MAP trajectory in some way, then the MAP trajectory can be misleading in our attempt to interpret observational data (see, e.g., [Lember and Koloydenko, 2014] and references therein). As a simple example, imagine the underlying state system is a series of independent flips of a biased coin that comes up heads with probability  $p > 0.5$ , and suppose we receive totally non-informative observations. Then the MAP trajectory is a series of all heads. But this path is “atypical” in a number of senses. For instance, we do expect some tails—in particular, about  $n(1-p)$  of them where  $n$  is the number of flips. This and a great many other statistics about the series of coin flips are not represented in the MAP path. This is not a criticism of Viterbi per se, but merely of the practice of thinking or hoping that the MAP path is somehow representative of other probable paths as well.

In some other work, the authors have pointed out problems with MAP paths for stochastic continuous-time discrete-state systems [Perkins, 2009, Levin et al., 2012]. Such systems dwell in a state for random period of time, before moving on to a randomly-chosen next state. Although MAP paths can be efficiently computed [Perkins, 2009], solutions are non-typical in that they involve “instant” transitions through low latency states and dwells in high latency

states. Similar issues can arise in discrete time HMMs, particularly, but not exclusively, when analyzing an HMM that arises from discretizing a continuous-time process.

A second problem with MAP paths is that they constitute an overly-specific inference. For example, in a simple speech recognition scenario, a MAP path might assign a word or phoneme to every timestep underlying the speech signal. However, transitions between words are not truly so crisp. Moreover, there is usually no point in assigning precise start and end times to each word or phoneme. If we hear, “The quick brown fox . . .”, what is the value of estimating that the word “The” ends and the word “quick” begins precisely 0.823 seconds into a spoken signal? It is the sequence of words spoken, and not their exact timing, that is important. Similarly for gesture recognition, activity recognition, etc.

Motivated by these criticisms of MAP trajectories, an alternative inference method, called *state sequence analysis*, was proposed earlier [Levin et al., 2012]. In this inference problem, the objective is to find the maximum a posteriori *sequence* of states, but averaging away (i.e. marginalizing over) the transition times between those states—essentially treating them as nuisance variables. In that work, however, the inference problem is solved for systems modeled as continuous-time Markov chains with initial and/or terminal probabilities; no observations during the time period of interest are allowed. In the present work, we study state sequence analysis for HMMs: the focus is on discrete time setting, where a series of noisy observations is allowed. We describe a provably-correct algorithm that finds the most probable state sequence given a trajectory of observations. Then, we demonstrate how state sequence analysis differs from Viterbi path inference on synthetic and real examples, and in particular (as it is designed to do) how state sequence analysis provides more accurate estimates of the sequence of states underlying a noisy series of observations.

## 2 BACKGROUND

Let  $\mathcal{X}$  be a discrete finite state space and  $\mathcal{O}$  the observation space of a hidden Markov model [Rabiner, 1989]. Let  $\mathbf{T}$  be the transition matrix of this HMM, with  $\mathbf{T}_{x,y}$  representing the probability of transitioning from state  $x \in \mathcal{X}$  to state  $y \in \mathcal{X}$ , and  $p_x(o)$  be the emission probability of observation  $o \in \mathcal{O}$  in state  $x \in \mathcal{X}$ .

For a possible trajectory of states visited by a discrete time HMM we are interested in identifying what is the corresponding duration-free sequence of states, i.e. sequence of states with self-loops removed. For example, given the trajectory of states  $\langle x, x, x, y, y, y, x \rangle$ , the corresponding *state sequence* will be  $\langle x, y, x \rangle$ . We denote the probability that HMM trajectory follows the state sequence  $\mathbf{s}$  given the se-

quence of  $n$  observations, as

$$P(X_{1:n} \in \text{seq}_n(\mathbf{s}) | \mathbf{o}_{1:n}),$$

where  $\text{seq}_n(\mathbf{s})$  is a set of all length  $n$  trajectories whose duration-free sequence equals to  $\mathbf{s}$ . Where possible, we will use  $P(\mathbf{s} | \mathbf{o}_{1:n})$  as a shortcut to the above notation.

### 3 THE STATE SEQUENCE INFERENCE PROBLEM

Finding the most probable state sequence, which we call *state sequence analysis* (SSA), can be seen as a search problem that requires evaluation of probabilities of state sequences. Recall that, to find the most probable trajectory of states, a naive exhaustive enumeration quickly becomes infeasible because the number of possible trajectories grows exponentially with the length of observation sequence. Similarly, a naive implementation of the search for the most probable state sequence does not scale well. Even a single evaluation of a probability of a state sequence involves the summation over possibly large number of terms. Specifically, the number of terms in the set  $\text{seq}_n(\mathbf{s})$  is equal to  $\binom{n-1}{|\mathbf{s}|-1}$  for the state sequence  $\mathbf{s}$ .

Nevertheless, inferring the most probable trajectory can be done efficiently by a well-known Viterbi algorithm that uses dynamic programming to do the search and evaluation simultaneously. Although a Viterbi-like approach does not appear to be possible to address the question we pose, in what follows we develop search and evaluation algorithms that make the problem tractable. First, we identify a particular structure within the search space that allows us to prune large parts of the space as the search progresses. Second, we provide a recursive relation that is used to efficiently evaluate probabilities of new state sequences using dynamic programming. However, prior to delving into the algorithmic details, we begin with a little discussion about the similarities and differences between most probable state sequence and most probable state trajectory.

#### 3.1 THE MOST PROBABLE STATE SEQUENCE AND TRAJECTORY

The definition of a state sequence presented in an earlier section leaves little doubt that the most probable state trajectory is not necessarily the same as the most probable state sequence, given a sequence of observations. Yet, to better understand the nature of their differences it might be helpful to pinpoint the cases where those two will, in fact, be equal. In the first case, consider a HMM in which probabilities of staying in any state are zero. Any state trajectory generated by this HMM will have no repetitive states, and therefore the only state sequences that have non zero probability of happening must be of the same length as the observation sequence. Hence, the most probable state tra-

jectory and the most probable state sequence will be the same.

In the second case, suppose that the observations identify the underlying hidden states exactly (observation sequence is also Markov). It implies that there is only one state trajectory that could generate a given sequence of observations. Hence, there is only one state sequence explaining a given observation sequence; that state sequence can be computed by collapsing repetitive states appearing in the only possible state trajectory.

Of course, both of these cases are rather extreme, and in their exact form can rarely be found in practice. However, one can expect that the ‘‘closer’’ the given HMM is to one of those extremes, the less evident will the difference be between the most probable state sequence and state trajectory (in its collapsed form). This intuition is also backed up, to some extent, by the experimental results presented in later sections.

#### 3.2 DOMINATION OF SEQUENCES

In [Levin et al., 2012], the authors develop a structural relation between state sequences of a continuous time Markov chain that allows them to avoid searching all of the space of sequences. In this section, we develop a similar relation between sequences in the setting of discrete time HMMs. The following derivation, which serves as a basis for this development, enables us to express the probability of longer state sequences in terms of shorter state sequences. Let  $\mathbf{s}_x$  be a state sequence that ends with state  $x$ , and let  $\mathbf{u}_y$  be a one step shorter subsequence of  $\mathbf{s}_x$  such that  $\mathbf{s}_x = \langle \mathbf{u}_y, x \rangle$ . Then,

$$\begin{aligned} P(\mathbf{s}_x | \mathbf{o}_{1:n}) &= \sum_{i=1}^{n-1} P[X_{1:i} \in \text{seq}_i(\mathbf{u}_y); X_{i+1:n} = x | \mathbf{o}_{1:n}] \\ &= \sum_{i=1}^{n-1} P[X_{1:i} \in \text{seq}_i(\mathbf{u}_y) | \mathbf{o}_{1:n}] \\ &\quad \cdot P[X_{i+1:n} = x | \mathbf{o}_{1:n}; X_i = y] \\ &= \sum_{i=1}^{n-1} \frac{P[\mathbf{o}_{1:n} | X_{1:i} \in \text{seq}_i(\mathbf{u}_y)] \cdot P[X_{1:i} \in \text{seq}_i(\mathbf{u}_y)]}{P(\mathbf{o}_{1:n})} \\ &\quad \cdot P[X_{i+1:n} = x | \mathbf{o}_{i+1:n}; X_i = y] \\ &= \sum_{i=1}^{n-1} \frac{P[\mathbf{o}_{1:i} | X_{1:i} \in \text{seq}_i(\mathbf{u}_y)] \cdot P[X_{1:i} \in \text{seq}_i(\mathbf{u}_y)]}{P(\mathbf{o}_{1:i})} \\ &\quad \cdot \frac{P[\mathbf{o}_{i+1:n} | X_i = y]}{P[\mathbf{o}_{i+1:n} | \mathbf{o}_{1:i}]} \\ &\quad \cdot P[X_{i+1:n} = x | \mathbf{o}_{i+1:n}; X_i = y] \\ &= \sum_{i=1}^{n-1} P[X_{1:i} \in \text{seq}_i(\mathbf{u}_y) | \mathbf{o}_{1:i}] \\ &\quad \cdot \frac{P[X_{i+1:n} = x; \mathbf{o}_{i+1:n} | X_i = y]}{P[\mathbf{o}_{i+1:n} | \mathbf{o}_{1:i}]} \\ &= \sum_{i=1}^{n-1} P(\mathbf{u}_y | \mathbf{o}_{1:i}) \cdot \frac{\mathbf{T}_{y,x} \mathbf{T}_{x,x}^{n-i-1} \prod_{j=i+1}^n p_x(o_j)}{P[\mathbf{o}_{i+1:n} | \mathbf{o}_{1:i}]} \end{aligned} \quad (1)$$

Eq. (1) makes the separation between the parameters of the problem (HMM, observation sequence) and the shorter state sequence explicit. Specifically, the probability of

longer state sequence equals to convolution of probabilities of shorter state sequence and the probability of staying in new state, normalized appropriately by the emission probabilities. As a result, the following definition of dominance between sequences turns out to be useful in speeding up the search.

**Definition 1.** Given a sequence of observations  $\mathbf{o}_{1:n}$ , let  $\mathbf{s}$  and  $\mathbf{v}$  be sequences that start from the same state and end with the same state. Then we say that  $\mathbf{s}$  dominates  $\mathbf{v}$ , and denote it by  $\mathbf{s} \gg_n \mathbf{v}$ , if

$$\forall i \in \{1, \dots, n\} : P(\mathbf{s} | \mathbf{o}_{1:i}) \geq P(\mathbf{v} | \mathbf{o}_{1:i}).$$

The dominance relation assures us that extensions of dominated sequences should never be explored within the search space, as the following lemma suggests.

**Lemma 1.** Let  $\mathbf{v}_y$  and  $\mathbf{s}_y$  be two state sequences that start with the same state and end with state  $y$ . If  $\mathbf{v}_y$  is dominated by  $\mathbf{s}_y$  given an observation sequence  $\mathbf{o}_{1:n-1}$  for some  $n > 1$ , then a one step extension of state sequence  $\mathbf{v}_y$  will be dominated by some other state sequence, given an observation sequence  $\mathbf{o}_{1:n}$ .

*Proof.* Observe that if  $\mathbf{v}_y$  is dominated by  $\mathbf{s}_y$  for an observation sequence  $\mathbf{o}_{1:n-1}$ , then the same holds for shorter observation sequences, e.g.  $\forall i \in \{1, \dots, n-1\} : \mathbf{o}_{1:i}$ , simply following the definition of the dominance. Now, let  $\langle \mathbf{v}_y, x \rangle$  be a one step extension of a state sequence  $\mathbf{v}_y$ . Since  $\mathbf{v}_y$  is dominated by  $\mathbf{s}_y$  for observation sequences  $\mathbf{o}_{1:i}$  ( $\forall i < n$ ), following equation (1) we get that

$$\forall i \leq n : P(\langle \mathbf{v}_y, x \rangle | \mathbf{o}_{1:n}) \leq P(\langle \mathbf{s}_y, x \rangle | \mathbf{o}_{1:n}).$$

□

Based on the above result, similarly to [Levin et al., 2012], we devise the algorithm that performs the search within the space of sequences by maintaining sets of non-dominated sequences only. The pseudo-code of the algorithm is provided in Algorithm 1 box. Each new sequence is checked against existing non-dominated sequences (line 6). If it appears to be non-dominated by any of those sequences, it is added to the set on non-dominated sequences (line 7). All sequences that are dominated by the new sequence in this set are removed (line 8), and all one step extensions of the new sequence will be checked by the algorithm later on (line 9).

### 3.3 COMPUTING THE PROBABILITY OF A STATE SEQUENCE

As mentioned above, performing the search within the space of state sequences will rely on evaluation of probabilities of those sequences. One approach to do so is based on implementing the recursive relation between sequences

given in Eq. (1) using dynamic programming. Following this route it requires  $O(n^2k)$  operations to evaluate the probability of a state sequence of length  $k$  given  $n$  observations. On the other hand, the following recursive relation can be used to evaluate probabilities in just  $O(nk)$  operations. As previously, let  $\mathbf{s}_x = \langle \mathbf{u}_y, x \rangle$  be a state sequence that consists of a shorter state sequence  $\mathbf{u}_y$  and a last state  $x$ .

$$\begin{aligned} P(\mathbf{s}_x | \mathbf{o}_{1:n}) &= \frac{P(\mathbf{o}_{1:n-1}, o_n | X_{1:n} \in \mathbf{seq}_n(\mathbf{s}_x)) \cdot P[X_{1:n} \in \mathbf{seq}_n(\mathbf{s}_x)]}{P(o_n | \mathbf{o}_{1:n-1}) \cdot P(\mathbf{o}_{1:n-1})} \\ &= \frac{P(\mathbf{o}_{1:n-1} | X_{1:n} \in \mathbf{seq}_n(\mathbf{s}_x)) \cdot P[X_{1:n} \in \mathbf{seq}_n(\mathbf{s}_x)]}{P(\mathbf{o}_{1:n-1})} \\ &\quad \cdot \frac{P(o_n | X_n = x)}{P(o_n | \mathbf{o}_{1:n-1})} \\ &= P[X_{1:n} \in \mathbf{seq}_n(\mathbf{s}_x) | \mathbf{o}_{1:n-1}] \\ &\quad \cdot \frac{p_x(o_n)}{P(o_n | \mathbf{o}_{1:n-1})}. \end{aligned} \quad (2)$$

Now, the first term in the product of Eq. (2) can be expanded as follows:

$$\begin{aligned} P[X_{1:n} \in \mathbf{seq}_n(\mathbf{s}_x) | \mathbf{o}_{1:n-1}] &= P[X_{1:n} \in \mathbf{seq}_n(\mathbf{s}_x); X_{n-1} = y | \mathbf{o}_{1:n-1}] \\ &\quad + P[X_{1:n} \in \mathbf{seq}_n(\mathbf{s}_x); X_{n-1} = x | \mathbf{o}_{1:n-1}] \\ &= P[X_{1:n-1} \in \mathbf{seq}_{n-1}(\mathbf{u}_y); X_n = x | \mathbf{o}_{1:n-1}] \\ &\quad + P[X_{1:n-1} \in \mathbf{seq}_n(\mathbf{s}_x); X_n = x | \mathbf{o}_{1:n-1}] \\ &= P(\mathbf{u}_y | \mathbf{o}_{1:n-1}) \mathbf{T}_{y,x} + P(\mathbf{s}_x | \mathbf{o}_{1:n-1}) \mathbf{T}_{x,x}. \end{aligned} \quad (3)$$

Combining Eq. (2) and Eq. (3) we get,

$$\begin{aligned} P(\mathbf{s}_x | \mathbf{o}_{1:n}) &= \frac{p_x(o_n)}{P(o_n | \mathbf{o}_{1:n-1})} \\ &\quad \cdot [P(\mathbf{u}_y | \mathbf{o}_{1:n-1}) \mathbf{T}_{y,x} + P(\mathbf{s}_x | \mathbf{o}_{1:n-1}) \mathbf{T}_{x,x}]. \end{aligned} \quad (4)$$

---

#### Algorithm 1 State Sequence Analysis for HMMs

---

- 1: Initialize  $\forall x, y : ND_{x,y} = \emptyset$  - sets of non-dominated sequences for each pair of start and end states  $x, y$
  - 2: Initialize Queue  $CheckQ = \{\forall x : \langle x \rangle, \forall x, y : \langle x, y \rangle\}$
  - 3: **while**  $CheckQ$  is not empty **do**
  - 4:     Fetch  $\mathbf{s}_{x,y} \in CheckQ$
  - 5:     Compute  $\forall i : P(\mathbf{s}_{x,y} | \mathbf{o}_{1:i})$
  - 6:     **if**  $\mathbf{s}_{x,y}$  is not dominated by  $ND_{x,y}$  **then**
  - 7:         Add  $\mathbf{s}_{x,y}$  to  $ND_{x,y}$
  - 8:         Remove all  $\mathbf{u}_{x,y} \in ND_{x,y}$  s.t.  $\mathbf{s}_{x,y} \gg_n \mathbf{u}_{x,y}$
  - 9:          $\forall z \neq y : \text{Add } \langle \mathbf{s}_{x,y}, z \rangle$  to  $CheckQ$
  - 10:     **end if**
  - 11: **end while**
  - 12: **return**  $\arg \max_{\forall x,y : \mathbf{s} \in ND_{x,y}} P(\mathbf{s} | \mathbf{o}_{1:n})$
-

## 4 EMPIRICAL EVALUATION

### 4.1 SYNTHETIC EVENT-DETECTION EXAMPLES

To demonstrate state sequence analysis, and its difference compared to the Viterbi algorithm, consider the HMM in Figure 1A. Trajectories begin in state  $S$  and can remain there through self-looping or move on either to state  $B$  or state  $E$ . State  $E$  is absorbing. When the system is in state  $B$  it can remain there through self-looping or proceed to state  $E$ . All states emit normal-distributed observations with standard deviation 1. However, the mean observation is zero in states  $S$  and  $E$ , and  $\mu_B$  in state  $B$ . Thus, states  $S$  and  $E$  appear the same, whereas state  $B$  may appear different if  $\mu_B \neq 0$ .

At a high level, this example models event detection problems—for instance, detecting a security intrusion [Qiao et al., 2002], detecting specific gestures [Dardas and Georganas, 2011], detecting molecular events [Schreiber and Karplus, 2015], etc. Essentially, there is a series of noisy but harmless or uninteresting events, punctuated, rarely and for a short time, with relevant activity. But that relevant activity may still be subtle to detect, depending on how different it is from the background.

We simulated state-observation trajectories of 100 steps from the HMM. Because the chance of following the self-loop on state  $S$  is 0.95, the chance that a trajectory remains in that state for all 100 steps is  $0.95^{99} \approx 0.0062$ . Thus, almost all trajectories move on from the initial state, and they do so on average after 20 steps. From  $S$ , trajectories are equally likely to proceed to  $E$  or  $B$ , thus approximately half of all trajectories will contain a visit to state  $B$ , and half will not. Because the self-loop probability on state  $B$  is  $2/3$ , trajectories remain there on average for just three steps before moving on to state  $E$ . Figure 1B shows example observation trajectories that respectively do not and do include a visit to state  $B$  with  $\mu_B = 4.5$ . In the second trajectory, the visit to  $B$  happens on steps 15 and 16, as indicated by the black bar below.

We simulated 10,000 state-observation trajectories, and for each observation series applied the Viterbi algorithm and state sequence analysis. We “collapsed” the simulated state trajectory into the sequence of states visited, and likewise for the Viterbi solution. Then, we counted on how many of the 10,000 simulations Viterbi and/or state sequence analysis inferred the correct state sequence from the observation series. Figure 1C shows the fraction of correct state sequence inferences for each algorithm as a function of  $\mu_B$ . When  $\mu_B = 0$ , there is no information to distinguish any of the states and so, unsurprisingly, both algorithms are right approximately half the time. Conversely, when  $\mu_B$  is large, a brief visit to state  $B$  is so obvious that the sequence is clear, and both algorithms get nearly 100% of

state sequence correct. In between, both algorithms have intermediate performance, but state sequence analysis is correct a greater fraction of the time. Figure 1D reports in greater detail the frequencies of simulated and inferred state sequences when  $\mu_B = 5$ . Both algorithms are correct more than 90% of the time, but state sequence analysis demonstrates substantially fewer “false positive” detections of state  $B$  than Viterbi does (66 versus 467), although it incurs a greater number of “false negatives” (inferring no  $B$  visit when there was one, 115 versus 36).

Figure 2A shows a similar problem where more than one event (visit to state  $B$ ) can occur in a given trajectory, with a sample observation trajectory displayed in Figure 2B with  $\mu_B = 4$ . Again, we ran 10,000 simulations of 100 time steps to generate observation trajectories, and then applied Viterbi and state sequence analysis to estimate the underlying state sequences. Figure 2C shows the results. Similar to the previous example, when  $\mu_B$  is high, so that events are clearly observed, both algorithms predict the number of events correctly. However, when  $\mu_B$  is lower, state sequence analysis outperforms Viterbi. Unlike the previous example, state sequence analysis predicts correctly more often than Viterbi even when  $\mu_B = 0$ , so that observations are uninformative. With uninformative observations, state sequence analysis predicts the sequence with the highest a priori probability, which turns out to be  $SBSBSBSBS$ . By contrast, Viterbi computes the maximum probability trajectory to be  $SS \dots S$ , but this corresponds to the state sequence  $S$ , which is of much lower probability. So, even with non-informative observations, state sequence analysis “guesses” correctly a greater fraction of the time. In part, this advantage extends to partially-informative observations. Figure 2D shows a heatmap of true numbers of events versus predicted numbers of events by the two algorithms, across our 10,000 simulations when  $\mu_B = 2$ . In this domain, we see that Viterbi tends to underestimate the number of events, whereas state sequence analysis appears much less biased.

### 4.2 ACTIVITY RECOGNITION DATASET

The increasing availability of various kinds of sensors allows us to collect and analyze data that was previously unthinkable. One place where sensors invade our lives is houses and apartments. The data collected from those sensors is the basis for designing various intelligent environments [Cook and Das, 2004, Augusto and Nugent, 2006] and several healthcare applications [Abowd et al., 2002, Suzuki et al., 2004]. In [van Kasteren et al., 2011], activity data was collected from several individuals living in an apartment/house. The data consists of sensor readings and activity annotations made either manually or automatically. The sensors installed around the house report, for example, open-close states of doors and cupboards, and pressure measurements on the couch, while the activities might in-

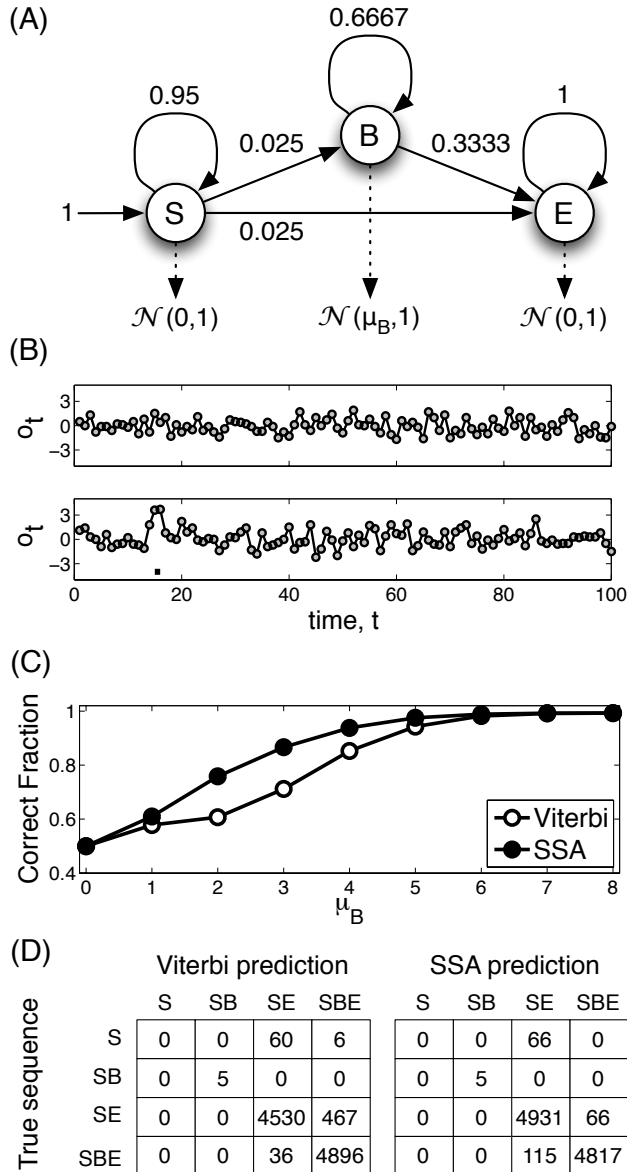


Figure 1: Demonstration of State Sequence Analysis, and comparison with the Viterbi algorithm, on a simple detection problem. Based on a noisy time series, the problem is to infer the underlying sequence of states, which is approximately equivalent to determining whether the sequence contains a visit to the state  $B$ .

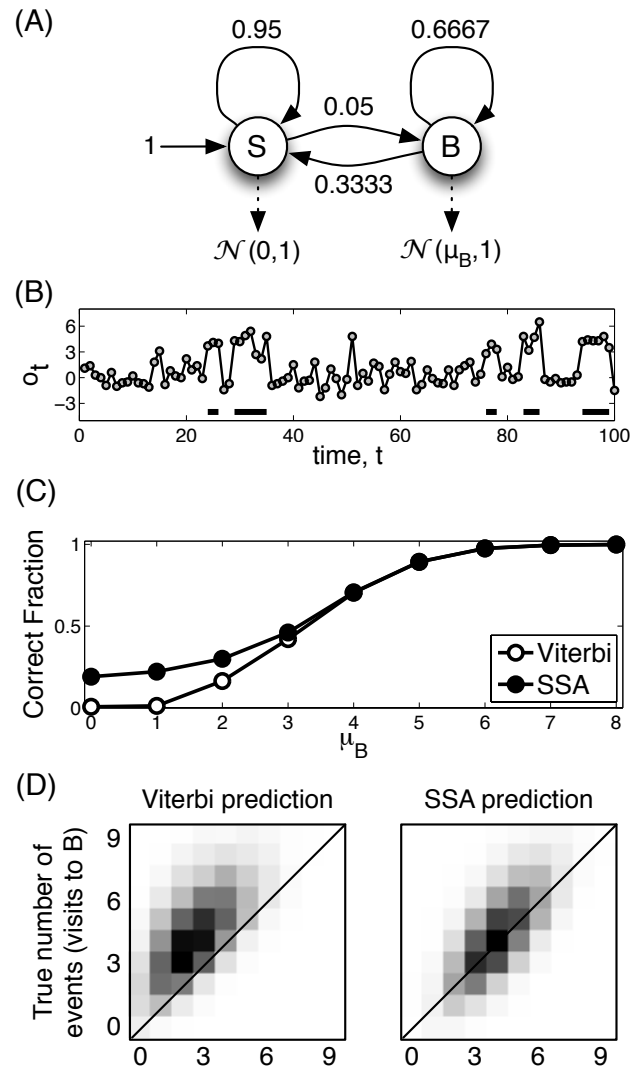


Figure 2: Comparison of State Sequence Analysis and Viterbi on a detection problem where multiple events per trajectory are possible.

clude cooking, eating, taking a shower, etc. The authors trained several models that identify past activities given a time series of sensor readings. Modeling the data with an HMM and using the Viterbi algorithm to identify the activities was among the options that demonstrated competitive recognition performance. The reported results offer a useful benchmark to evaluate new methods of activity recognition in this setting.

Interestingly, the current problem formulation for activity recognition task around the house does not necessarily reflect the type of questions that the user is interested to find answers to. In particular, it is easy to imagine that what really matters to the user is the sequence of activities performed in a period of time, regardless of their duration. If an intelligent system is able to identify the activity perfectly at each point of time, one can obtain the sequence of activities simply by collapsing repetitive activities into a single value. Clearly, sensor readings do not always help to identify activities perfectly. As seen from the synthetic problem described earlier, when observations provide limited information about the underlying state, collapsing repetitive activities identified by the most probable state trajectory (Viterbi output) is often inferior to finding the most probable state sequence directly.

To illustrate the potential benefits of state sequence analysis on a realistic problem, we evaluate it on the largest out of three available datasets collected in [van Kasteren et al., 2011]. This dataset, named *House A*, contains 592 hours (nearly 25 days) of recorded activities and sensor readings of a single individual living in a one floor apartment. There are 14 sensors with binary outputs reporting different states of objects and 10 possible activities that the person can do at any single time period (see Table 1). A single record of activity and sensor readings was recorded every minute. To compare Viterbi and SSA approaches, 25 HMMs were trained, each on data from different sets of 24 days. These HMMs were then used, in combination with Viterbi or SSA, to make predictions on 24 trajectories (one hour each) taken from the remaining single day. For more details about the dataset see [van Kasteren et al., 2011].

As previously, we compared the performance of SSA to the results obtained by collapsing repetitive activities in the most probable state trajectory computed using the Viterbi algorithm. Out of 592 sample trajectories, Viterbi and SSA disagreed on 25 ( $\approx 5\%$ ) of those. Further, we measured how well the produced state sequences match the true state sequences using the insert-delete string comparison<sup>1</sup>. On 7 trajectories the results of Viterbi produced a better score, while on 13 trajectories SSA performed better (for the other

<sup>1</sup>This is similar to the well known edit/Levenshtein string distance, except that the replace operation is not directly allowed. In the context of activity recognition problems, it is better not to report a particular activity at all than to report a wrong activity.

|   |
|---|
| Activities: <i>idle, leave house, use toilet, take shower, brush teeth, go to bed, prepare breakfast, prepare dinner, get snack, get drink.</i> |
|---|

|   |
|---|
| Sensors: <i>microwave, hall-toilet door, hall-bathroom door, cups cupboard, fridge, plates cupboard, front door, toilet flush, freezer, pans cupboard, groceries cupboard, hall-bedroom door.</i> |
|---|

Table 1: House A: activities and sensors.

5 the scores were equal). Although the difference might appear not very meaningful at first glance, there are several factors about the problem that need to be understood. First, some sensors identify the activities rather precisely, e.g., using toilet flush indicates that the toilet was used at that time, or opening the front door means that the person is leaving/coming back to the apartment. Second, most of the true state sequences are short (see Table 2, first line). In fact, around 75% of those contain only a single state, the majority of which are either *idling* (unidentified activity), *go to bed* or *leave house*. However, hours that include 3 activities or more are much more likely to produce sequences of observations on which Viterbi and SSA differ (see Table 2, second line). Considering that, it is expected to have a small number of trajectories on which Viterbi and SSA disagree. Among those, SSA provided a better prediction on nearly twice the number of trajectories on which Viterbi performed better. An example of a one hour trajectory where SSA improves over Viterbi inference is given in Figure 3. This is a typical example where sensor readings provide only indirect information about an activity that happens, which Viterbi algorithm fails to identify. From the computational perspective, it took about 5 minutes to run SSA algorithm on the entire dataset using a 3.40GHz CPU with 16GB memory machine, and a fraction of a second to find the most probable state trajectories (Viterbi).

To further establish the fact that SSA performs better as compared to Viterbi algorithm on this domain, we trained an HMM on the entire dataset and sampled a large number of state and observation trajectories. Specifically, for each starting state of an HMM we sampled 1000 one-hour-long trajectories and evaluated SSA and Viterbi on this simulated dataset. The results are presented in Figure 4. As expected, depending on the initial state of the HMM, there will be a different number of trajectories on which SSA and Viterbi state sequences are not equal. Moreover, depending on the initial state, the general performance of SSA and Viterbi, compared to each other, can be different. Nevertheless, in most cases SSA performed significantly better than Viterbi. In total, Viterbi scored better on 286 trajectories while SSA scored better on 494 trajectories, which amounts to more than 70% improvement in performance.



| length $\geq 1$ | length $\geq 2$ | length $\geq 3$ | length $\geq 4$ |
|-----------------|-----------------|-----------------|-----------------|
| 100% (592)      | $\approx 25\%$  | $\approx 20\%$  | $\approx 12\%$  |
| 100% (25)       | 96%             | 92%             | 52%             |

Table 2: House A: the percentage of state sequences of different length appearing in the dataset (line 1), and those whose Viterbi and SSA scores were different (line 2).

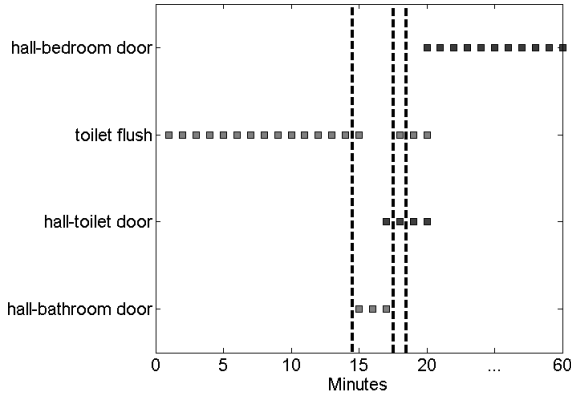


Figure 3: Example of a one hour trajectory of observations. Y axis enumerates binary sensor readings, and points on the plot identify active sensors. Note that we use the sensor readings representation that continues to be active until there is any change in other sensor readings (see more details in [van Kasteren et al., 2011]). Vertical dashed lines on the plots show the time of true activity change. The true underlying activity sequence is: *idle*, *brush teeth*, *use toilet*, *go to bed*. SSA outputs the true activity sequence, whose probability equals to 0.53. However, Viterbi’s output omits the *brush teeth* activity, and the probability of resulting activity sequence is only 0.34.

## 5 CONCLUSION

Inference problem in Hidden Markov Models have received considerable attention, and several algorithms have been used in a variety of domains to infer the behavior of the underlying system this HMM represents, given a sequence of noisy observations. Typically, these algorithms estimate different quantities involving hidden variables (e.g., Viterbi, posterior decoder, most probable annotation sequence, etc. [Brejová et al., 2007, Lember and Koloydenko, 2014]) and therefore, at their core, address different inference problems. In this work, we point out that in a variety of domains none of these techniques is adequate enough to answer our question of interest, which is to find the most probable state sequence. Naturally, the closest existing approach that can be used to find the most probable state sequence is to find the most probable state trajectory using the Viterbi algorithm and collapse repetitive state visitations. However, by doing so, the recovered state sequence is not guaranteed to be the

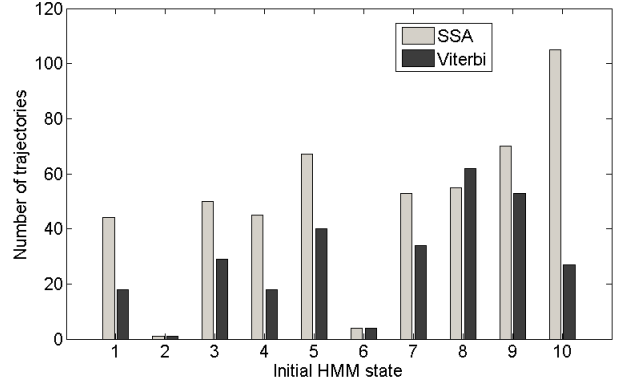


Figure 4: Comparison of SSA and Viterbi evaluated on simulated trajectories from *House A* dataset. X axis identifies the initial state of an HMM that the trajectories were sampled from. For each state, 1000 trajectories were sampled, and the performance of SSA was compared to Viterbi. As previously, trajectories on which the outputs of SSA and Viterbi differ were used to measure the performance by means of insert-delete string distance (Viterbi/SSA vs. true state sequence). Each bar in a category represents the number of trajectories on which the corresponding method scored better.

most probable state sequence. Moreover, the resulting state sequence might actually have a much lower probability of happening compared to the most probable one, as our experimental results suggest. In fact, those discrepancies are not surprising since, using Bayesian networks terminology, the Viterbi algorithm for HMMs produces *most probable explanation* (MPE) solution, while we are seeking to find a MAP solution different from MPE as we ignore dwelling time (variables) [Darwiche, 2009].

Building on earlier work where an algorithm to find the most probable state sequence in continuous time Markov chains was proposed [Levin et al., 2012], in this work we developed a state sequence analysis algorithm that finds the most probable state sequence of an HMM given a sequence of observations. The algorithm performs a search in the space of state sequences by evaluating those sequences and pruning parts of the search space by carefully maintaining a domination relationship between the sequences. We evaluated the algorithm on synthetic event-detection problems first to highlight its advantages over a Viterbi-based approach. Then, we used state sequence analysis to find the most probable sequence of activities based on a real activity recognition dataset collected from individuals performing different duties at home [van Kasteren et al., 2011]. Although the Viterbi algorithm performed reasonably well on this problem, the advantages of using state sequence analysis were nevertheless apparent.

Much remains to be done with respect to the analysis and evaluation of our proposed approach. One of the topics that

requires further attention is the computational complexity of the algorithm. Although we did not experience difficulties running the algorithm on the problems presented in this paper, it is still not clear whether the problem of finding the most probable state sequence is polynomial or NP hard in general. Further, in many real dynamical systems that involve hidden variables, the model of choice is Hidden Semi-Markov Models (HSMM) [Yu, 2010]. It seems rather straightforward to adapt our algorithm to HSMMs, however experimental results are needed to verify its benefits and computational cost.

## Acknowledgements

This work was supported in part by a Discovery grant from the Natural Sciences and Engineering Research Council of Canada (NSERC) to TJP, and by an NSERC Postdoctoral Fellowship to YG.

## References

- [Abowd et al., 2002] Abowd, G. D., Bobick, A. F., Essa, I. A., Mynatt, E. D., and Rogers, W. A. (2002). The aware home: A living laboratory for technologies for successful aging. In *Proc. of the AAAI Workshop "Automation as Caregiver"*, pages 1–7.
- [Augusto and Nugent, 2006] Augusto, J. C. and Nugent, C. D. (2006). *Designing smart homes: the role of artificial intelligence*, volume 4008. Springer Science & Business Media.
- [Bahl et al., 1986] Bahl, L., Brown, P., de Souza, P. V., and Mercer, R. (1986). Maximum mutual information estimation of hidden Markov model parameters for speech recognition. In *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing.*, volume 11, pages 49–52.
- [Blei and Moreno, 2001] Blei, D. M. and Moreno, P. J. (2001). Topic segmentation with an aspect hidden Markov model. In *Proc. of ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 343–348. ACM.
- [Brejová et al., 2007] Brejová, B., Brown, D. G., and Vinař, T. (2007). The most probable annotation problem in HMMs and its application to bioinformatics. *Journal of Computer and System Sciences*, 73(7):1060–1077.
- [Cook and Das, 2004] Cook, D. and Das, S. (2004). *Smart environments: Technology, protocols and applications*, volume 43. John Wiley & Sons.
- [Dardas and Georganas, 2011] Dardas, N. H. and Georganas, N. D. (2011). Real-time hand gesture detection and recognition using bag-of-features and support vector machine techniques. *IEEE Trans. on Instrumentation and Measurement*, 60(11):3592–3607.
- [Darwiche, 2009] Darwiche, A. (2009). *Modeling and reasoning with Bayesian networks*. Cambridge University Press.
- [Haussler and Eeckman, 1996] Haussler, D. K. D. and Eeckman, M. G. R. F. H. (1996). A generalized hidden Markov model for the recognition of human genes in DNA. In *Proc. Int. Conf. on Intelligent Systems for Molecular Biology*, pages 134–142.
- [Lember and Koloydenko, 2014] Lember, J. and Koloydenko, A. A. (2014). Bridging viterbi and posterior decoding: a generalized risk approach to hidden path inference based on hidden Markov models. *The Journal of Machine Learning Research*, 15(1):1–58.
- [Levin et al., 2012] Levin, P., Lefebvre, J., and Perkins, T. J. (2012). What do molecules do when we are not looking? state sequence analysis for stochastic chemical systems. *Journal of The Royal Society Interface*, 9(77):3411–3425.
- [Nguyen et al., 2005] Nguyen, N. T., Phung, D. Q., Venkatesh, S., and Bui, H. (2005). Learning and detecting activities from movement trajectories using the hierarchical hidden Markov model. In *Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 955–960. IEEE.
- [Perkins, 2009] Perkins, T. J. (2009). Maximum likelihood trajectories for continuous-time Markov chains. In *Advances in Neural Information Processing Systems*, pages 1437–1445.
- [Qiao et al., 2002] Qiao, Y., Xin, X., Bin, Y., and Ge, S. (2002). Anomaly intrusion detection method based on HMM. *Electronics Letters*, 38(13):663–664.
- [Rabiner, 1989] Rabiner, L. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. of the IEEE*, 77(2):257–286.
- [Schreiber and Karplus, 2015] Schreiber, J. and Karplus, K. (2015). Analysis of nanopore data using hidden Markov models. *Bioinformatics*, Epub ahead of print: Feb. 3.
- [Sonnhammer et al., 1998] Sonnhammer, E. L., Von Heijne, G., Krogh, A., et al. (1998). A hidden Markov model for predicting transmembrane helices in protein sequences. In *Proc. Int. Conf. on Intelligent Systems for Molecular Biology*, pages 175–182.
- [Suzuki et al., 2004] Suzuki, R., Ogawa, M., Otake, S., Izutsu, T., Tobimatsu, Y., Izumi, S.-I., and Iwaya, T. (2004). Analysis of activities of daily living in elderly people living alone: single-subject feasibility study. *Telemedicine Journal & E-Health*, 10(2):260–276.
- [van Kasteren et al., 2011] van Kasteren, T., Englebienne, G., and Kröse, B. J. (2011). Human activity recognition from wireless sensor network data: Benchmark and software. In *Activity recognition in pervasive intelligent environments*, pages 165–186. Springer.
- [Wang et al., 2007] Wang, K., Li, M., Hadley, D., Liu, R., Glessner, J., Grant, S. F., Hakonarson, H., and Bucan, M. (2007). PennCNV: an integrated hidden Markov model designed for high-resolution copy number variation detection in whole-genome SNP genotyping data. *Genome research*, 17(11):1665–1674.
- [Yamato et al., 1992] Yamato, J., Ohya, J., and Ishii, K. (1992). Recognizing human action in time-sequential images using hidden Markov model. In *Proc. IEEE Int. Conf. on Vision and Pattern Recognition*, pages 379–385.
- [Yu, 2010] Yu, S.-Z. (2010). Hidden semi-markov models. *Artificial Intelligence*, 174(2):215–243.

---

# Multitasking: Efficient Optimal Planning for Bandit Superprocesses

---

Dylan Hadfield-Menell and Stuart Russell

## Abstract

A bandit superprocess is a decision problem composed from multiple independent Markov decision processes (MDPs), coupled only by the constraint that, at each time step, the agent may act in only one of the MDPs. Multitasking problems of this kind are ubiquitous in the real world, yet very little is known about them from a computational viewpoint, beyond the observation that optimal policies for the superprocess may prescribe actions that would be suboptimal for an MDP considered in isolation. (This observation implies that many applications of sequential decision analysis in practice are technically incorrect, since the decision problem being solved is often part of a larger, unstated bandit superprocess.) The paper summarizes the state-of-the-art in the theory of bandit superprocesses and contributes a novel upper bound on the global value function of a bandit superprocess, defined in terms of a direct relaxation of the arms. The bound is equivalent to an existing bound (the Whittle integral), but is defined constructively, as the value of a related multi-armed bandit. We provide a new method to compute this bound and derive the first practical algorithm to select optimal actions in bandit superprocesses. The algorithm operates by repeatedly establishing dominance relations between actions using upper and lower bounds on action values. Experiments indicate that the algorithm's run-time compares very favorably to other possible algorithms designed for more general factored MDPs.

## 1 INTRODUCTION

Multitasking is no doubt an activity familiar to the reader: one faces several decision problems but can act on only one (or perhaps a bounded number) at a time. Such prob-

lems are ubiquitous for individuals, corporations, armies, and governments.

Multitasking problems are expressed by the class of *bandit superprocesses* [Nash, 1973] or BSPs. A  $k$ -armed BSP  $M$  consists of  $k$  independent Markov decision processes  $M_1, \dots, M_k$ ; the MDPs are coupled by a common discount factor and by the constraint that at each time step the agent can act in only one MDP.

Since the MDPs are independent of each other, one might imagine that the optimal policy for  $M$  is obtained by solving each MDP—turning the BSP into a multi-armed bandit—and then using Gittins indices to choose a sequence of arms. In fact, *the optimal policy for a BSP may include actions that are suboptimal from the point of view of the constituent MDP in which they are taken*. The reason for this is that the availability of other MDPs in which to act changes the balance between short-term and long-term rewards in a component MDP; in fact, it tends to lead to greedier behavior in each MDP because aiming for long-term reward in one MDP would delay rewards in all the other MDPs. The globally and locally optimal policies necessarily coincide only when the discount factor is 1.

Hence, in addition to their general importance in the real world, a second reason to study multitasking problems is that they undermine an assumption implicit in practical applications of sequential decision analysis: the assumption that an optimal solution for the user's decision problem is optimal for a user who faces multiple decision problems.

Despite these considerations, there has been remarkably little research on bandit superprocesses and almost none in AI. Section 2 summarizes what is known. Obviously, there are connections to multi-armed bandits (MABs), which are a special case of BSPs in which each arm only allows one choice of action rather than several. Unfortunately, the index theorems that simplify the computation of optimal MAB policies are not valid for BSPs. There are also strong connections—hitherto unexplored—between BSPs and *sums of games* as studied in combinatorial game theory [Conway, 2000]. The principal question we address in

this paper is whether an algorithm exists that is substantially more efficient than applying a standard MDP solver to the “cross-product” MDP obtained by combining the states spaces of the constituent MDPs.

The sole known case where planning for a BSP is provably more efficient is the case where a *dominating* policy exists: if a single policy is optimal across the family of *retirement processes* associated with each arm then a BSP can be reduced to an equivalent MAB [Whittle, 1980]. (Retirement processes provide a measure of how the optimal policy depends on context and are defined in Section 3.) However, this condition is seldom satisfied in practice.

This paper provides three contributions. First, we give a concise survey of the bandit superprocess literature tailored to the AI community. Second, we provide a novel upper bound on the global value function of a BSP. For a BSP,  $M$ , with arms  $\{M_1, \dots, M_k\}$  we relax each arm to obtain an MAB  $M' = \{M'_1, \dots, M'_k\}$  by adding actions to ensure that dominating policies exist. We show that this upper bound is equivalent to the *Whittle integral*, an existing upper bound for BSPs [Brown and Smith, 2013]. Because our bound is defined in terms of an explicit relaxation, it provides insight into the nature of the bound and opens an avenue to extend this work to more general MDPs. Finally, we describe a practical computational approach for solving BSPs: we derive a simple method for computing the Whittle integral upper bound that can use an arbitrary MDP solver as a black box; then we combine this upper bound with a lower bound to derive an efficient algorithm for  $\epsilon$ -optimal decision making in a BSP. We present empirical results to show that it substantially improves over more general optimal factored MDP algorithms; we use it to compute provably optimal actions for problems with upwards of  $10^{30}$  states in the full state space.

## 2 RELATED WORK

Robbins [1952] provided the first formulation of multi-armed bandits (MABs) in their modern form. The famous Gittins index theorem [Gittins, 1979] showed that optimal MAB policies are obtained by ranking the arms according to an index function defined on each separately. An immediate corollary is that optimal decision making in an MAB is *linear* in the number of arms. Problems with this property are said to be *indexable*.

Bandit superprocesses (BSPs) were introduced by Nash [1973] as a generalization of multi-armed bandits to study allocation of resources among research projects. Whittle [1980] provided an alternate proof of the Gittins index theorem that extends to bandit superprocesses with dominating policies. His proof utilized a construction called the *Whittle integral*, which allows one to compute the value of a composite state in an MAB. Glazebrook [1982] provides a proof that bandit superprocesses are *not* indexable

in general. Glazebrook [1993] considers bandit superprocesses where the arms can exert limited influence on each other and shows a result analogous to Whittle’s.

Brown and Smith [2013] were the first to identify the significance of the Whittle integral for sequential decision making. They developed a version of policy iteration to compute it, and recognized that it upper bounds the value function of a BSP,<sup>1</sup> but did not derive an algorithm for solving BSPs. Our work provides two further contributions regarding the Whittle integral: a short proof that it is an upper bound and a simpler algorithm to compute it.

Within the AI community, there has been limited study of loosely coupled Markov decision processes. Singh and Cohn [1998] consider optimal solutions to *simultaneous MDPs*, where an agent can take actions in a number of MDPs. Their formulation is more general than ours, as it is possible to act in multiple MDPs at once. They derive bounds for this problem and give an algorithm that combines a form of real-time dynamic programming with pruning steps to remove provably suboptimal actions. However, their bounds are substantially looser than ours, and our experiments show that the corresponding algorithm can be impractical for simple BSPs with many arms.

Meuleau et al. [1998] examine MDPs that are coupled by constraints on the use of shared resources. BSPs fit within this class if we view the restriction to a single MDP at a time as a constraint on the agent’s attention. Interestingly, their heuristics are also defined in terms of a parameterized value function for the component MDPs—it would be interesting to attempt to generalize the approaches considered here to their problem domain. In this work, we leverage the particular structure of our resource to compute optimal solutions; Meuleau et al. construct a heuristic policy.

## 3 TECHNICAL BACKGROUND

### MARKOV DECISION PROCESSES

**Definition 1.** (*Markov Decision Process [Puterman, 2009]*) A (finite-state, discounted) MDP,  $M$ , is a tuple  $M = \langle \mathcal{S}, \mathcal{A}, T, R, \gamma \rangle$ .  $\mathcal{S}$  is a set of states.  $\mathcal{A}$  is a set of actions.  $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$  is a function that assigns probability to state transitions for each state–action pair.  $R$  is a (bounded) reward function that maps state–action pairs to (positive) rewards  $R : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}^+$ .  $\gamma \in [0, 1)$  is a discount factor.

A solution to  $M$  is a policy,  $\pi$ , that maps states to actions. The value of a state,  $s$ , under  $\pi$  is the sum of expected discounted rewards received by starting in  $s$  and selecting ac-

<sup>1</sup>This result first appears in the literature as an intermediate step in a larger proof from Whittle [1980]. It went unnoticed or unappreciated in the intervening 30+ years.

tions according to  $\pi$ :

$$V^\pi(s) = \mathbb{E} \left[ \sum_{t=0}^{\infty} \gamma^t R(s_t) | s_0 = s, \pi \right].$$

The optimal policy,  $\pi^*$ , maximizes this value. In the above definition, and the ones that follow, we use superscripts to indicate dependence on the agent’s policy. To simplify notation, we will omit these superscripts when the policy referred to is the optimal policy (e.g.,  $V(s) = V^{\pi^*}(s)$ ). The  $Q$ -function for the state–action pair,  $(s, a)$ , is the value of taking  $a$  in  $s$  and selecting future actions according to  $\pi^*$ .

## RETIREMENT PROCESSES

Given an MDP, Whittle defines a family of optimal stopping problems:

**Definition 2.** (*Retirement Process [Whittle, 1980]*) Let  $M$  be an MDP. For  $\rho \geq 0$ , the retirement process for  $M$  with retirement reward,  $\rho$ , is an MDP,  $M_\rho$ , with a single additional state,  $s_R$ , and action,  $a_R$ .  $a_R$  transitions deterministically to  $s_R$  and receives reward  $\rho$ .  $s_R$  is a sink state that accrues zero reward.

We refer to a decision to select  $a_R$  as a decision to retire. We denote the retirement process value function as a function of a state and retirement reward,  $V(s, \rho)$ . We let the optimal policy for retirement reward  $\rho$  be  $\pi_\rho^*$ . We write the set of states where the policy,  $\pi$ , retires as  $\tau_\rho^\pi$ . We use  $\rho_-(\pi)$  and  $\rho_+(\pi)$  to denote the interval of retirement rewards so that  $\pi$  is optimal:

$$\rho' \in [\rho_+(\pi), \rho_+(\pi)] \Rightarrow V^\pi(s, \rho) = V(s, \rho).$$

We adopt the convention from the MAB/BSP literature and abuse notation to denote the (random) number of steps prior to retirement as  $\tau_\rho^\pi(s)$ . For  $s' \in \tau_\rho^\pi$  we let  $P_{retire}(s'|s, \rho, \pi)$  be the probability that  $s'$  is the first state in  $\tau_\rho^\pi$  the agent will reach given that it is in state  $s$  and executes policy  $\pi$ . We denote the expected discounted reward accrued prior to retirement, starting in  $s$ , as  $R_\rho^\pi(s)$ . In regions of retirement reward where the optimal policy and stopping rule do not change,  $\frac{\partial V}{\partial \rho}(s, \rho)$  is defined and is equal to the expected value of the discount parameter at retirement. This allows us to write the following expression for the retirement process value function:

$$V(s, \rho) = R_\rho(s) + \mathbb{E}[\gamma^{\tau_\rho(s)}] \rho. \quad (1)$$

$V(s, \rho)$  is piecewise linear in  $\rho$ . Figure 1 shows  $V(s, \rho)$  for the example BSP in Ex. 1. A policy that is optimal for every setting of  $\rho$  is called a *dominating policy*.

**Definition 3.** (*Dominating Policy*) Let  $\pi$  be a policy for an MDP,  $M$ .  $\pi$  is a dominating policy iff

$$\forall \rho \geq 0 \forall s \notin \tau_\rho \pi(s) = \pi_\rho^*(s).$$

## MULTI-ARMED BANDITS AND BANDIT SUPERPROCESSES

Multi-armed bandits (MAB) are a restricted class of MDPs that have received extensive study. Of particular interest are the form of the optimal policy and its utility in modelling “exploration vs exploitation” trade-offs.

An MAB consists of a set of Markov reward processes (MRPs), where an MRP is an MDP with a single action. Each MRP is referred to as an *arm* of the problem. At each time-step, the agent selects an arm, that arm transitions according to its transition distribution, and the agent receives the corresponding reward. We adopt a summation notation to indicate the combination of several MRPs into an MAB. For example, if  $X$  and  $Y$  are MRPs and  $Z$  is the MAB with arms  $X$  and  $Y$ , we will write  $Z = X + Y$ .

A famous result is that the optimal policy for any MAB is an *index policy* [Gittins, 1979]. Each state,  $s_i$ , in the individual arms is assigned an index,  $I_{s_i}$ . In joint state  $s = \{s_1, \dots, s_k\}$ , the optimal action is to select  $\text{argmax}_i I_{s_i}$ . For MAB arm  $M_i$  in state  $s_i$ , the index is defined as the value of retirement reward such that the agent is indifferent between immediate retirement and following the optimal stopping policy [Whittle, 1980]:

$$I_{s_i} = \min_{\rho} \{ \rho ; V_i(s_i, \rho) = \rho \}.$$

This means that, in a sense, context for a multi-armed bandit can always be summarized by a single number.

To model a multi-tasking problem, we consider a generalization of multi-armed bandits: bandit superprocesses (BSPs). Bandit superprocesses allow arms that are arbitrary MDPs (and so are a generalization of MABs); at each time step, the agent selects both an arm *and* an action to take within that arm.

**Definition 4.** (*Bandit Superprocess [Nash, 1973]*) Given  $k$  MDPs,  $\{M_i = \langle \mathcal{S}_i, \mathcal{A}_i, T_i, R_i, \gamma \rangle\}$ , we define

$$M = \sum_i M_i = \langle \mathcal{S}, \mathcal{A}, T, R, \gamma \rangle$$

to be the bandit superprocess with arms  $\{M_i\}$ .  $\mathcal{S} = \times_i \mathcal{S}_i$  and  $\mathcal{A} = \cup_i \mathcal{A}_i$ . The transition distribution is stationary for arms that are not selected and follows the identical reward and transition distributions for the selected arm.

Naturally, this makes planning more difficult. To see how, consider the following proposal:

**Conjecture 1.** Let  $X = \langle \mathcal{S}_X, \mathcal{A}_X, T_X, R_X, \gamma \rangle$  be an MDP. Define  $Y$  similarly and let  $Z = X + Y$  be the bandit superprocess that is their sum. Let  $a \in \mathcal{A}_X$ ,  $s_X \in \mathcal{S}_X$  be a state–action pair from  $X$ . Suppose that this transition is suboptimal in every retirement process:  $\forall \rho \geq 0, V_X(s_X, \rho) > Q_X((s_X, a), \rho)$ . Then a is

suboptimal for any state in  $Z$  with  $s_X$  as one of the components:

$$\forall s_Y \in \mathcal{S}_Y, V_Z(\{s_X, s_Y\}) > Q_Z(\{s_X, s_Y\}, a).$$

This conjecture essentially proposes that state–action pairs can be safely ignored if they are suboptimal for all settings of a constant alternative. Unfortunately, as the following example illustrates, this is not the case.

**Example 1.** Define  $X$  to be a deterministic reward chain and  $Y$  to be an initial choice between three reward chains  $(y_0, y_1, y_2)$  defined as follows:

$$X = 28, 28, 28, 28, 28, 28, 0, 0, \dots$$

$$Y = \begin{cases} y_0 = 100, 100, 100, 0, 0, 0, \dots \\ y_1 = 99, 99, 99, 1.4, 1.4, 1.4, \dots \\ y_3 = 28, 28, 28, \dots \end{cases}$$

The retirement process value function for each reward chain can be computed by simulating the Gittins index policy:<sup>2</sup>

$$V(y_0, \rho) = \max \left\{ 100 \left( \sum_{t=0}^2 \gamma^t \right) + \gamma^3 \rho, \rho \right\}$$

$$V(y_1, \rho) = \max \left\{ 99 \left( \sum_{t=0}^2 \gamma^t \right) + \gamma^3 \max \left\{ \frac{1.4}{1-\gamma}, \rho \right\}, \rho \right\}$$

$$V(y_2, \rho) = \max \left\{ \frac{28}{1-\gamma}, \rho \right\}.$$

If we let  $\gamma = .9$ , then we have

$$\forall 280 > \rho \geq 0 \max\{V(y_0, \rho), V(y_2, \rho)\} > V(y_1, \rho).$$

Thus, any policy for a retirement process retirement process derived from  $Y$  that initially selects  $y_1$  is suboptimal. We can apply the same strategy to compute the value of each combination of reward streams:

$$V(y_0 + X) = 100 \left( \sum_{t=0}^2 \gamma^t \right) + \gamma^3 \left( 28 \sum_{t'=0}^5 \gamma^{t'} \right)$$

$$V(y_1 + X) = 99 \left( \sum_{t=0}^2 \gamma^t \right) + \gamma^3 \left( 28 \sum_{t'=0}^5 \gamma^{t'} + \gamma^6 \frac{1.4}{1-\gamma} \right)$$

$$V(y_2 + X) = \frac{28}{1-\gamma}$$

From this we can see that the optimal policy for the BSP that combines these MDPs,  $Z = X + Y$ , initially collects reward from  $y_1$ . This contradicts Conjecture 1.

<sup>2</sup>Here we use the fact that the Gittins index of a non-increasing reward sequence is equal to the instantaneous reward.

## 4 AN UPPER BOUND FOR BANDIT SUPERPROCESSES

In this section we show a bound on the value function of a bandit superprocess. We derive this bound by adding actions to a BSP so that it is equivalent to an MAB. The value function of our relaxed BSP is equivalent to the Whittle integral bound derived in Brown and Smith [2013] and so it yields insight into that computation. We begin by defining the Whittle integral and show some basic results in order to motivate our relaxation.

**Definition 5.** (Whittle Integral [Brown and Smith, 2013]) Let  $M$  be a BSP. Let  $i$  index the arms of  $M$ . For any state,  $s = \{s_i\}$ , and  $\rho \geq 0$ , the Whittle integral of  $s$  is defined as

$$\hat{V}(s, \rho) = I - \int_{x=\rho}^I dx \prod_i \frac{\partial V_i}{\partial \rho}(s_i, x). \quad (2)$$

Where  $I \geq \max_i I_{s_i}$ .

When the arms of a BSP admit a dominating policy the Whittle integral is equal to the value function:

**Theorem 1.** (Whittle Condition [Whittle, 1980]) Let  $M$  be a  $k$ -armed BSP with components  $\{M_i\}$  and state space  $\mathcal{S}$ . If each  $M_i$  has a dominating policy, then

$$\forall s \in \mathcal{S}, \forall \rho \geq 0, \hat{V}(s, \rho) = V(s, \rho).$$

MRPs have a single action per state, so the Whittle condition is trivially satisfied for all multi-armed bandits.  $V(s, 0) = V(s)$ , so  $\hat{V}(s, 0)$  provides an efficient method to compute the value of an MAB. For BSPs an arm that satisfies the Whittle condition can be replaced with an MRP that selects actions according to  $\pi^*$ .

The formula in Eq. 2 lends itself to a straightforward implementation, but is very challenging to interpret. We show below that this is equivalent to evaluating the retirement process value function for a single arm with a set of rewards determined by the other arms<sup>3</sup>. We refer to this set of rewards as the *critical points* of those arms.

**Definition 6.** (Critical Points of an MDP) Let  $M$  be a Markov decision process. The critical points of  $M$ ,  $\mathcal{C}(M) = \{\rho_i\}$ , are the values of retirement reward such that the optimal stopping rule or policy changes.

These are points where there is a discontinuity in  $\frac{\partial V}{\partial \rho}$ . We let

$$\Delta^M(s, \rho) = \lim_{\delta \rightarrow 0} \frac{\partial V_M}{\partial \rho}(s, \rho + \delta) - \frac{\partial V_M}{\partial \rho}(s, \rho - \delta)$$

be the size of the corresponding discontinuities. This is equivalent to the expected increase in  $\mathbb{E}[\gamma^\tau]$  under the new stopping rule. Theorem 2 shows that  $\Delta$  and  $\mathcal{C}$  characterize the interaction between arms of an MAB.

<sup>3</sup>Our result is a small extension on a related result in Brown and Smith [2013]; it is primarily included to provide intuition.

**Theorem 2.** Let  $X, Y$  be Markov reward processes. Let  $Z = X + Y$  be the 2-armed bandit.  $\forall s = \{s_X, s_Y\} \in \mathcal{S}_Z$ ,

$$V_Z(s) = \sum_{\rho \in \mathcal{C}(Y)} V_X(s_X, \rho) \Delta^Y(s_Y, \rho). \quad (3)$$

*Proof.* (sketch) As a first step, we follow the steps in Whittle [1980] and integrate Equation 2 by parts. This expresses  $\hat{V}$  as the following integral:

$$\hat{V}_Z(s) = \int_{\rho'=\rho}^{I_{s_X}} V_X(s_X, \rho') \frac{\partial^2 V_Y(s_Y, \rho')}{\partial \rho^2} d\rho'.$$

$V_Y$  is piecewise linear with respect to  $\rho$  so  $\frac{\partial^2 V_Y}{\partial \rho^2}$  is a weighted sum of delta functions centered at  $V_Y$ 's kinks.  $\mathcal{C}(Y)$  and  $\Delta^Y$  respectively characterize these kinks and weights. Thus,  $\hat{V}_Z(s)$  is equal to the rhs of Eq. 3.  $X$  and  $Y$  are MRPs, so an appeal to Theorem 1 shows the result.  $\square$

A similar property holds for arbitrary  $k$ -armed bandits. This shows that we can summarize the context for an arm as a collection of weighted retirement rewards. Turning to BSPs, this lends insight into the approximation the Whittle integral introduces.

To see how, we reconsider Ex. 1. Recall that this example consists of the BSP  $Z = X + Y$  and that  $Y$  is a choice between three reward chains,  $\{y_0, y_1, y_2\}$ . Theorem 2 allows us to write the gap between the Whittle integral upper bound and the value of selecting  $y_1$  as

$$\sum_{\rho \in \mathcal{C}(X)} \left( \max_i V_{y_i}(s_i, \rho) - V_{y_1}(s_1, \rho) \right) \Delta^X(s_X, \rho)$$

Figure 1 shows the retirement process value functions for this example. While the retirement process value of  $y_1$  is always less than that of either  $y_0$  or  $y_2$ , it is close enough to their maximum that choosing  $y_1$  essentially achieves the upper bound. The Whittle integral for  $Z$  is a weighted combination of distinct retirement process value functions ( $V_{y_0}, V_{y_2}$ ) but, in reality, the agent will be forced to pick a single policy of the two.

## DOMINATED RELAXATION OF AN MDP

In this section, we introduce our primary theoretical result: a relaxation for the arms of a BSP so that a dominating policy exists. This reduces the BSP to an MAB whose value upper bounds the value of states in the BSP. We can show that the Whittle integral computes the value of states in this MAB and arrive at a straightforward proof that the Whittle integral is an upper bound. We call the result the *dominated relaxation* of an MDP. Before providing the details, we illustrate the main ideas with an example. The relaxed MDP is actually a *Semi-MDP* (SMDP): a generalization of an MDP where each action,  $a$ , has a duration,  $\delta(a) \in \mathbb{R}_+$ .

**Example 2.** Let MDP  $M$  be an initial choice between MRPs:

$$M = \begin{cases} 15, 0, 0, 0 \dots \\ 10, 10, 10, 3, 3, 3, \dots \end{cases}.$$

Let  $T(B)$  be the top (bottom) MRP. Let  $\pi_T$  ( $\pi_B$ ) be the policy that selects  $T(B)$  in  $s_0$ . We can relax  $M$  with the addition of a single durative action,  $a'$ , that transitions from the sink state in  $T$  to the sink state  $B$ . We set  $\delta(a') = 2$ . We take  $\rho$  such that  $V^{\pi_T}(s_0, \rho) = V^{\pi_B}(s_0, \rho)$  and set the rewards associated with  $a'$  to be

$$R_{\rho}^{\pi_B}(s_0) - R_{\rho}^{\pi_T}(s_0) = 10 \sum_0^2 \gamma^t - 15 = 12.1.$$

With this change, at low settings of retirement reward, the agent is indifferent between a policy that opts for the top chain then selects  $a'$  and the bottom reward chain. For high settings of retirement reward, the optimal policy retires immediately or retires after collecting the reward of 15. This SMDP satisfies the Whittle condition and can be replaced by an MRP in any bandit superprocess. Fig. 1 (c) shows an illustration of the state space and the introduced action.

In this example, we connected the state where  $\pi_T$  retires to the state where  $\pi_B$  retires. This lets the agent collect short-term and long-term rewards with the same policy. To do this in general, we introduce multiple copies of the state space, one for each policy that is optimal for some  $\rho$ .

**Definition 7.** (*Dominated Relaxation of an MDP*) Let  $M$  be an MDP with discount factor  $\gamma$  and state space  $\mathcal{S}$ . Let  $s$  be a state in  $M$ . The dominated relaxation of  $M$  for  $s$ ,  $M_D(s)$ , is a semi-Markov decision process that fixes  $s$  as an initial state. Let  $\{\pi_i\}$  be the policies that are optimal for some  $\rho$ :  $\{\pi_{\rho}^* | \rho \in \mathcal{C}(M)\}$ . This sequence is ordered so that  $\rho_-(\pi_i)$  is increasing in  $i$ <sup>4</sup>.

For each  $i$ , we introduce a copy of the state space,  $\mathcal{S}_i$ , where the agent is restricted to following  $\pi_i$ . Let  $s'_i$  be the analogue of  $s'$  in  $\mathcal{S}_i$ . For  $s'_i \in \tau_{\rho_-(\pi_i)}$ , we introduce a single durative action,  $a_i$ , that takes the agent from  $\mathcal{S}_i$  to  $\mathcal{S}_{i-1}$  and characterize it as follows:

- $R(s'_i) = R_{\rho_+(\pi_{i-1})}(s) - R_{\rho_-(\pi_i)}(s)$
- $T(s'_i, a_i, s''_{i-1}) = P_{retire}(s'' | \pi_{i-1}, \rho_+(\pi_{i-1}), s)$
- $\delta(a_i) = \log_{\gamma} \mathbb{E} \left[ \gamma^{\tau_{\rho_+}^{\pi_{i-1}}(s)} \right] - \log_{\gamma} \mathbb{E} \left[ \gamma^{\tau_{\rho_-}^{\pi_i}(s)} \right]$

For each  $i$ , we introduce an action that transitions from  $s$  to  $s_i$  with  $\delta = 0$ . Let  $V_D(s)$  represent the value of  $s$  in  $M_D(s)$ .

**Theorem 3.** Let  $M$  be an MDP with state space  $\mathcal{S}$ . The following statements are true for  $s \in \mathcal{S}$  and  $\rho \geq 0$ :

<sup>4</sup>Recall that  $[\rho_-(\pi), \rho_+(\pi)]$  is the interval of retirement rewards where  $\pi$  is optimal.

1.  $M_D(s)$  satisfies the Whittle condition.
2.  $V_D(s, \rho) = \hat{V}(s, \rho)$ .
3.  $\hat{V}(s, \rho) \geq V(s, \rho)$

*Proof.* See supplementary materials □

## 5 AN $\epsilon$ -OPTIMAL ALGORITHM FOR BANDIT SUPERPROCESSES

In this section, we present two algorithms related to BSPs. The first is a novel algorithm to compute  $V(s, \rho)$  that can use any method to solve the underlying MDP. The second is an efficient algorithm to compute optimal actions for a BSP. Our approach, *Branch-and-Bound Value Iteration* (BBVI), uses Whittle integrals to compute upper and lower bounds on value. Then, we apply a modified Branch-and-Bound search to find provably optimal actions.

### COMPUTING A RETIREMENT PROCESS VALUE FUNCTION

Before we present BBVI, we give a simple algorithm to compute a retirement process value function that uses an (arbitrary) MDP solver as a black box. Brown and Smith [2013] describe an algorithm to compute retirement process value functions, but their approach requires custom implementation of a modified simplex algorithm. Our approach is simple and based on an algorithm that initially appeared in the solutions manual for Russell and Norvig [2010].

Our goal is to identify each component of  $V(\cdot, \rho)$ , so our output will be a list of critical points and the associated slopes. First, we will need the following result.

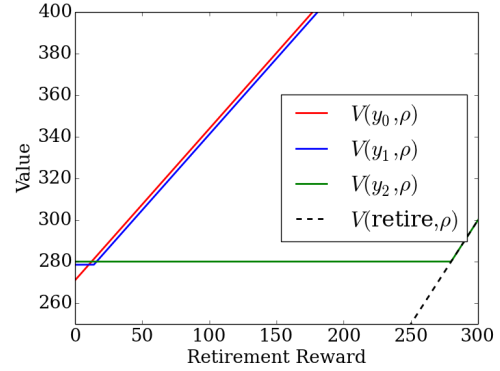
**Lemma 1.** *Let  $M$  be an MDP with state space  $\mathcal{S}$ . Let  $\rho_0 < \rho_1$ . Consider a retirement reward in the interior of this interval,  $\rho \in (\rho_0, \rho_1)$ . If,  $\forall s \in \mathcal{S}$*

$$V(s, \rho) = V(s, \rho_0) + \frac{V(s, \rho_1) - V(s, \rho_0)}{\rho_1 - \rho_0}(\rho - \rho_0), \quad (4)$$

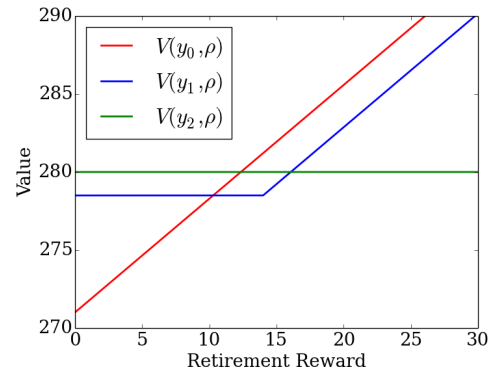
*then there is at least one policy and stopping rule that is optimal for every  $\rho' \in [\rho_0, \rho_1]$ . The expected value of the discount factor at retirement is the slope between those points:*

$$\mathbb{E} \left[ \gamma^{\tau^*(s)} \right] = \frac{V(s, \rho_1) - V(s, \rho_0)}{\rho_1 - \rho_0}. \quad (5)$$

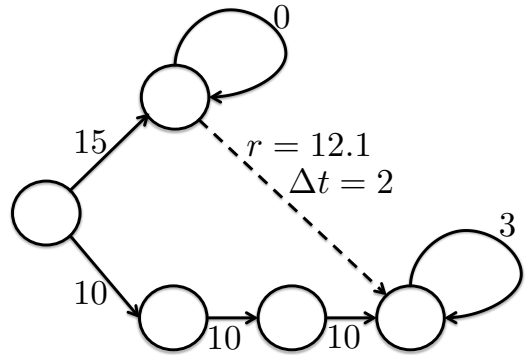
*Proof.* This follows from the form of Equation 1 and the fact that  $V(s, \rho)$  is piecewise linear, increasing, and convex. □



(a)



(b)



(c)

Figure 1: (a-b) Retirement process value functions for the reward chains in Ex. 1. The slope of the lines is the expected value of the discount parameter at retirement:  $\mathbb{E}[\gamma^\tau]$ . Flat sections indicate regions of retirement rewards where retirement is always suboptimal. The kink in the green curve at  $\rho = 280$  indicates that it has become optimal to retire immediately. The kink in the black curve at  $\rho = 14$  increases the slope to  $\gamma^6 < 1$ ; for  $\rho > 14$ , it is optimal to retire after collecting a prefix of the reward stream from  $y_2$ . (c) Dominated relaxation of the MDP in Ex. 2. We add additional durative actions to the state space to ensure that a dominating policy exists. Note that the original MDP does not admit a dominating policy.



This test allows us to implement a binary search over retirement rewards. Our approach relies on a given MDP solution method, SOLVE, that returns the vector of values for a given MDP. We initialize an interval that contains all critical points and make a call to SOLVE to compute the retirement process value at each endpoint. Then we call SOLVE to compute the retirement process value for the midpoint of our interval and apply the test in Eq. 4. If it succeeds, we return the lower endpoint and the corresponding slope. If this does not, we sub-divide our interval and recurse. We can concatenate the results to get the breakpoints and slopes for the retirement process value function over this interval. Algorithm 1 shows pseudocode for this algorithm.

---

**Algorithm 1** Computing an MDP’s Critical Points

---

**Define:** CRITICALPOINTS( $M[\rho^-, \rho^+, V^-, V^+]$ )  
**Input:** MDP,  $M$ ; Interval of retirement values,  $\rho^-, \rho^+$ ; Values at interval endpoints,  $V^-, V^+$   
**if**  $\rho^-, \rho^+$  are not set **then**  
     $\rho^- \leftarrow 0$   
     $\rho^+ \leftarrow \frac{\text{maxReward}(M)}{1-\gamma}$   
     $V^- \leftarrow \text{SOLVE}(M_{\rho^-})$   
     $V^+ \leftarrow \rho^+ \quad \text{## Retirement is initially optimal}$   
**end if**  
     $\rho \leftarrow \frac{\rho^+ - \rho^-}{2}$   
     $V \leftarrow \text{SOLVE}(M_{\rho})$   
**if**  $|V - (V^- + (\rho - \rho^-) \frac{V^+ - V^-}{\rho^+ - \rho^-})| < \epsilon$  **then**  
    **return**  $[\rho^-], \left[ \frac{V^+ - V^-}{\rho^+ - \rho^-} \right]$   
**else**  
     $pts^-, slope^- \leftarrow \text{CRITICALPOINTS}(M, \rho^-, \rho, V^-, V)$   
     $pts^+, slope^+ \leftarrow \text{CRITICALPOINTS}(M, \rho, \rho^+, V, V^+)$   
    **## Merge adjacent intervals with the same slope**  
    **if**  $slope^-[-1] = slope^+[0]$  **then**  
         ~~$pts^+[0]$~~ ,  ~~$slope^+[0]$~~   
    **end if**  
    **return**  $pts^- \cup pts^+, slopes^- \cup slopes^+$   
**end if**

---

**BRANCH-AND-BOUND VALUE ITERATION**

Now, we present a practical algorithm to compute optimal actions in a bandit superprocess. While a BSP is not indexable, we would like to be able to plan efficiently when the arms are ‘close’ to indexable—when there are only a few states where the optimal policy changes in response to the opportunity cost of delayed rewards in other arms.

Our approach is based on envelope dynamic programming methods to solve MDPs: we compute value estimates for a given initial state by solving a dynamic program defined over a reachable subset of the state space [Gardiol, Natalia H and Kaelbling, Leslie P, 2003, Hansen and Zilberstein, 2001]. The primary difference between our approach and standard envelope methods is that we use dynamic programming

on our envelope to update an upper bound and a lower bound on the value. This is useful in a BSP because the Whittle integral allows efficient calculation of both bounds.

Our goal is to compute the optimal action for a given state  $s$  in a BSP,  $M = \langle M_1, \dots, M_K \rangle$ . We can obtain an upper bound on  $V(s)$  with a Whittle integral. We use a Whittle integral for the MAB that solves each arm independently to compute a lower bound. Our algorithm maintains upper and lower bounds on the  $Q$ -function for each action that could be executed from  $s$ . We write these bounds as  $Q^+(s, a)$  and  $Q^-(s, a)$  respectively. If there is a pair of actions  $a, a'$  such that  $Q^+(s, a') < Q^-(s, a) + \epsilon$ , then we can conclude that  $a'$  is at least  $\epsilon$ -suboptimal. If this test removes all but a single action, then we have found a  $\epsilon$ -optimal action and return it.

In the event that more than one action remains, we do a partial expansion of the state space around  $s$ . We keep track of a set of expanded states,  $\mathcal{E}$ , and a set of boundary states,  $\mathcal{B}$ . States in the boundary set,  $\mathcal{B}$ , are states that some expanded state can transition to but have not yet been added to  $\mathcal{E}$ . We can use these states to update the bounds on  $Q(s, \cdot)$  by solving a related MDP. This is formalized in the following theorem.

**Theorem 4.** *Let  $M$  be an MDP with state space  $\mathcal{S}$  and action space  $\mathcal{A}$ . Let  $\mathcal{S}' \subseteq \mathcal{S}$  where  $\mathcal{S}' = \mathcal{B} \cup \mathcal{E}$  and  $\forall s \in \mathcal{E}, T(s, a, s') \neq 0 \Rightarrow s' \in \mathcal{S}'$ . Let  $M^+$  be an MDP with state space  $\mathcal{S}' \cup \{\alpha\}$ .  $M^+$ ’s transition distribution and rewards are identical for states in  $\mathcal{E}$  (expanded states). Each state  $s \in \mathcal{B}$  transitions deterministically to  $\alpha$  and receives a reward that is an upper bound on  $V_M(s)$ .  $\alpha$  is a sink state that accrues 0 reward. Then the value of a state in  $M^+$  is an upper bound on its value in  $M$ :*

$$V_{M^+}(s) \geq V_M(s).$$

*Proof.* See supplementary materials. □

It is straightforward to show that an updated lower bound can be computed by fixing boundary states to have value equal to a lower bound. Our approach alternates between pruning actions based on dominance relations, adding states to  $\mathcal{E}$ , and computing the value of an MDP defined over  $\mathcal{E}$ . We interleave value iteration with a branch-and-bound search, so we call it Branch-and-Bound Value Iteration (BBVI). Algorithm 2 shows pseudocode to implement this method. At termination, the action we return is guaranteed to be at most  $\epsilon$ -suboptimal.

**Theorem 5.** *Let  $M$  be a BSP and let  $s$  be a state in  $M$ . Let  $a$  be the action returned by BBVI( $M, s, \epsilon$ ).*

$$V(s) - Q(s, a) < \epsilon.$$

*Proof.* See supplementary materials. □

Large regions of the state space are irrelevant to our objective so we use a heuristic to guide node expansion. Our heuristic is a measure of the sensitivity of values at the root to change in the upper and lower bounds of unexpanded nodes. Similar heuristics have been applied in many settings [Rivest, 1987, Smith and Simmons, 2004]. In BBVI, these values are the expected discounted visitation rates in the bounding MDPs and are computed as the dual variables from a value iteration LP. The backup procedure is slower than node expansions, so we perform backups in batches. In between backups and heuristic computations, we approximate the state space as a tree and push new states onto the agenda with their parent’s heuristic value weighted by the transition probability.

---

**Algorithm 2** Branch-and-Bound Value Iteration

---

**Define:** BBVI( $\langle M_0, \dots, M_K \rangle, s_0, \epsilon$ )

**Input:** BSP arms,  $M_k$ ; Initial state,  $s_0$ ; Tolerance,  $\epsilon$

# Lower bound  $M$  by fixing a policy for each arm  
 $LB_k \leftarrow \text{toMRP}(M_k)$   
 Compute critical points for  $M_k, LB_k$   
 Compute bounds on  $Q(s_0, \cdot)$  with Whittle integrals for  $M$  and  $LB$ .

# Keep track of expanded region of state space  
 $\mathcal{E} \leftarrow \emptyset$

# Keep track of states at boundary of  $M^+, M^-$   
 $\mathcal{B} \leftarrow \{s_0\}$   
 $a^* \leftarrow \operatorname{argmax}_a Q^-(s_0, a)$

**while**  $\exists a' \neq a^* \text{ s.t. } Q^+(s_0, a') \geq Q^-(s_0, a^*)$  **do**  
    $s \leftarrow \text{pop}(\mathcal{B})$   
    $\mathcal{E} \leftarrow \mathcal{E} \cup \{s\}$   
   **for**  $a \in \mathcal{A}$  **do**  
     **for**  $s' \in \text{successors}(s)$  **do**  
       Compute upper and lower bounds for  $Q(s', \cdot)$   
        $\mathcal{B} \leftarrow \mathcal{B} \cup \{s'\}$   
     **end for**  
   **end for**  
    $Q^+ \leftarrow \text{SOLVE}(\text{BOUNDMDP}(\mathcal{E}, \mathcal{B}, Q^+))$   
    $Q^- \leftarrow \text{SOLVE}(\text{BOUNDMDP}(\mathcal{E}, \mathcal{B}, Q^-))$   
    $a^* \leftarrow \operatorname{argmax}_a Q^-(s_0, a)$

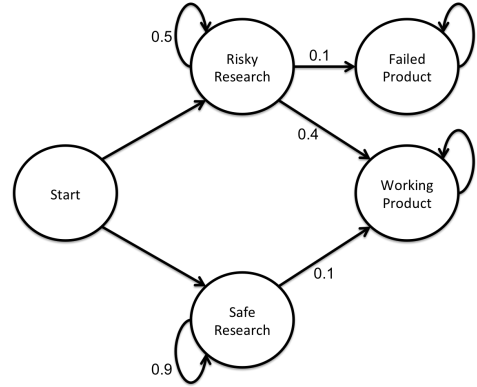
**end while**  
**return**  $\epsilon$ -optimal action  $a^*$

---

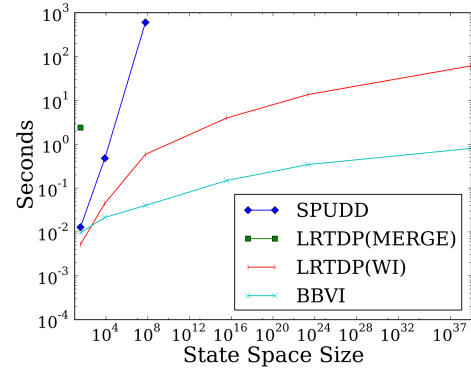
## 6 EMPIRICAL EVALUATION

We implemented BBVI in Python and use the linear programming solver Gurobi to compute value functions. Our experiments were run on an Intel i7 with 16 GB of RAM. In our first experiments, we compare the scalability of BBVI with that of standard MDP algorithms. In particular we compare the following algorithms:

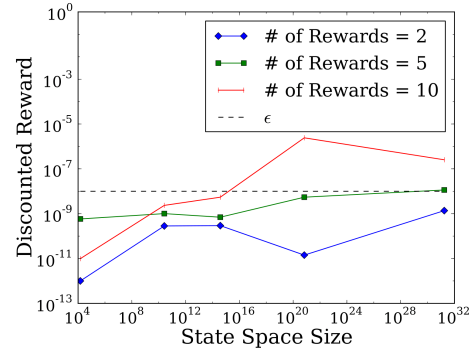
- SPUDD: the baseline factored MDP algorithm in the 2011 IPPC [Hoey et al., 1999].



(a) State Space for R & D BSP



(b) Runtime vs. Problem Size



(c) Solution Quality vs. Problem Size

Figure 2: (a) the state space for an arm of a R & D BSP. All states accrue 0 reward except for the ‘Working Product’ state, which collects a fixed reward per time step. (b) shows runtime (log-scale) vs problem size for BBVI and several alternatives on an R & D BSP. SPUDD [Hoey et al., 1999] and LRTDP (with the heuristic from Singh and Cohn [1998]) scale poorly compared with algorithms that leverage the Whittle integral. BBVI’s additional improvement over LRTDP stems from the use of an efficient, exact lower bound to check convergence. (c) BBVI’s bound on suboptimality after 10,000 node expansions (measured in units of discounted reward). Changing the number of rewards in each arm allows us to measure BBVI’s performance as arms become more sensitive to context.

- LRTDP(Merge): Labeled Real-Time Dynamic Programming [Bonet and Geffner, 2003] with the upper bound from Singh and Cohn [1998] as a heuristic.
- LRTDP(WI): LRTDP with the Whittle integral as an upper bound.
- BBVI: Branch-and-Bound Value Iteration.

We evaluate performance on a simple BSP designed to model allocation of research resources to product research and development. Each arm in this problem corresponds to a potential product that our agent could sell in the market. However, before we sell a product we must devote effort to R&D. This can be done in one of two ways, a safe research approach that is slow but reliable, and a risky approach that is fast, but runs a risk of producing a defective product.

After at least one product has been researched, the agent can opt to spend a round to produce and sell that product or continue research on a different project. Because taking a product to market does not change the state of the BSP, it is straightforward to show that any stationary policy (including  $\pi^*$ ) will only ever produce a single product. We define a distribution over these problems by selecting a random market value for each product uniformly from  $[0, 1]$  and random durations for the safe and risky research strategies (although we ensure that safe research is at least 3x as slow as risky research). Figure 2 (a) shows the state space for an example arm with typical parameters.

Figure 2 (b) shows the results of our comparison. We used a timeout of 1000s and set a memory limit of 4 GB. We ran our experiments in an online setting and running times reflect the amount of time to select an optimal action from scratch. While this is a natural setting for LRTDP and BBVI, it is admittedly a little unfair to SPUD (which computes a full policy). This is mitigated by the fact that BBVI's improvement over SPUD, which is a little under 5 orders of magnitude with  $10^8$  states, is such that an agent would need to be planning over an immense horizon before SPUD presents a reasonable alternative.

The factored MDP bounds from Singh and Cohn [1998] are quite ill-suited to this problem. This is because its upper bound (the sum of the independent value for each arm) is very loose and it essentially forces LRTDP to enumerate the state space. In contrast, LRTDP performs quite well when it has good heuristic information. However, its performance degrades faster than that of BBVI, because it does not effectively leverage a lower bound on value. Even for very large problem instances, BBVI computes optimal solutions in well under a second.

BBVI's performance in the R&D domain is largely explained by the structure of the arms and the short horizon within each arm. Although there may be a large number of arms, BBVI reduces each individual arm to an equivalent MRP after a small number of node expansions. This

means that running time is essentially linear in the number of arms. Note that, in this domain, the solution that solves each MDP independently and executes the corresponding index policy will just opt for conservative research on the most valuable option. This policy is essentially always sub-optimal: as long as there are reasonable alternatives it is usually a good idea to try some risky projects.

Our next experiment uses a synthetic domain to explore the performance of BBVI as the structure and number of arms changes. In this BSP, each arm is a 20x20 grid world. The actions in this world correspond to moving in the 4 cardinal directions. Rewards are 0 everywhere except at randomly chosen locations and after receiving a reward, the agent transitions to a random location.

In this experiment we vary the number of rewards that are available in each arm. Adding rewards will typically cause the optimal policy for an arm to be more sensitive to context. This means that the dominated relaxation will produce a looser upper bound, because it will need to add more actions. We evaluate this difficulty by measuring the convergence criterion of BBVI after 10,000 node expansions. Figure 2 (c) shows the results of this experiment. We can see that BBVI's solution quality decreases with the number of states, but the primary variation comes from changes in the properties of the arms.

## 7 CONCLUSION AND FUTURE WORK

In summary, we presented a model of highly decoupled decision making: bandit superprocesses. A BSP presents an agent with the opportunity to act in one of several Markov decision processes. The key constraint is that the agent can only act in a single process at a time. We provided a summary of BSP research, including an upper bound for the value function of a BSP in the form of the Whittle integral. We derived an equivalent relaxation and thus gave a novel proof that the Whittle integral is an upper bound. Finally we presented and evaluated algorithmic solutions for this class of decision problems.

In future work, we plan to extend these ideas in three directions. The first is algorithmic improvements for BSPs. A potential direction here is to leverage factored dynamics in the bounding MDPs themselves. A search over sequences of MDPs that only consider states where the current policy stops could be more efficient. Next, also plan to explore generalizations of the bounds used for BSPs to more general cases of global resource constraints (e.g., those considered in Meuleau et al. [1998]). Finally, the similarity to sums of games studied in Conway [2000] suggests an unexplored connection between BSPs and combinatorial game theory.

## References

- Blai Bonet and Hector Geffner. Labeled RTDP: Improving the convergence of real-time dynamic programming. In *IEEE Conference on Automated Planning and Scheduling*, pages 12–21, 2003.
- David B. Brown and James E. Smith. Optimal sequential exploration: Bandits, clairvoyants, and wildcats. *Operations Research*, 61(3):644–665, 2013.
- John H. Conway. *On Numbers and Games*. CRC Press, 2000.
- Gardiol, Natalia H and Kaelbling, Leslie P. Envelope-based planning in relational MDPs. In *Advances in Neural Information Processing Systems*, page None, 2003.
- John C. Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 148–177, 1979.
- Kevin D. Glazebrook. On a sufficient condition for super-processes due to Whittle. *Journal of Applied Probability*, pages 99–110, 1982.
- Kevin D. Glazebrook. Indices for families of competing Markov decision processes with influence. *The Annals of Applied Probability*, pages 1013–1032, 1993.
- Eric A Hansen and Shlomo Zilberstein. Lao: A heuristic search algorithm that finds solutions with loops. *Artificial Intelligence*, 129(1):35–62, 2001.
- Jesse Hoey, Robert St-Aubin, Alan Hu, and Craig Boutilier. SPUDD: Stochastic planning using decision diagrams. In *Fifteenth Conference on Uncertainty in Artificial Intelligence*, pages 279–288. Morgan Kaufmann Publishers Inc., 1999.
- Nicolas Meuleau, Milos Hauskrecht, Kee-Eung Kim, Leonid Peshkin, Leslie Pack Kaelbling, Thomas L Dean, and Craig Boutilier. Solving very large weakly coupled Markov decision processes. In *Fifteenth National Conference on Artificial Intelligence*, pages 165–172, 1998.
- Peter Nash. *Optimal allocation of Resources Between Research Projects*. PhD thesis, University of Cambridge, 1973.
- Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 2009.
- Ronald L. Rivest. Game tree searching by min/max approximation. *Artificial Intelligence*, 34(1):77–96, 1987.
- Herbert Robbins. Some aspects of the sequential design of experiments. In *Herbert Robbins Selected Papers*, pages 169–177. Springer, 1952.
- Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. 3 edition, 2010. Exercise 17.5.
- Satinder Singh and David Cohn. How to dynamically merge Markov decision processes. *Advances in Neural Information Processing Systems*, (10):1057–1063, 1998.
- Trey Smith and Reid Simmons. Heuristic search value iteration for POMDPs. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pages 520–527. AUAI Press, 2004.
- Peter Whittle. Multi-armed bandits and the Gittins index. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 143–149, 1980.

---

# Importance Sampling over Sets: A New Probabilistic Inference Scheme

---

**Stefan Hadjis, Stefano Ermon**  
Department of Computer Science  
Stanford University, Stanford, CA, USA  
{shadjis, ermon}@cs.stanford.edu

## Abstract

Computing expectations in high-dimensional spaces is a key challenge in probabilistic inference and machine learning. Monte Carlo sampling, and importance sampling in particular, is one of the leading approaches. We propose a generalized importance sampling scheme based on randomly selecting (exponentially large) subsets of states rather than individual ones. By collecting a small number of extreme states in the sampled sets, we obtain estimates of statistics of interest, such as the partition function of an undirected graphical model. We incorporate this idea into a novel maximum likelihood learning algorithm based on cutting planes. We demonstrate empirically that our scheme provides accurate answers and scales to problems with up to a million variables.

## 1 INTRODUCTION

Probabilistic inference is one of the key computational challenges in statistical machine learning and Bayesian statistics. The key computational bottleneck for many statistical inference problems of interest lies in the computation of high-dimensional integrals. Examples include computing posterior probabilities, model averaging, and evaluating partition functions of undirected graphical models. The field is dominated by two main paradigms tracing their roots to statistics and physics: Monte Carlo sampling methods [1, 15, 19] and variational techniques [16, 29]. Monte Carlo sampling is an extremely influential idea leveraged by numerous algorithms which have found an enormous number of applications in many fields of scientific computation, with application ranging from machine learning, statistics, and physics. It is often ranked among the most important algorithms of all time in polls and surveys [4]. The basic idea is to estimate properties of a high-dimensional space (e.g., the integral of a function) by look-

ing at a small number of representative states. The major difference between Monte Carlo schemes lies in how these representative states are selected and weighted.

Importance sampling (IS) is one of the most popular Monte Carlo sampling schemes. It is a simple and elegant idea, which is at the core of other widely used techniques such as Markov Chain Monte Carlo, Annealed Importance Sampling [21], and others [10]. The approach is very general: one can choose the samples randomly according to any desired *proposal distribution* (some mild restrictions have to be met), and IS provides a recipe to properly weight the samples and obtain an estimate for the original high-dimensional integral. The choice of the proposal distribution affects the variance of the estimate, and the number of samples required to obtain a statistically reliable estimate can grow exponentially in the problem size if the proposal distribution is poorly chosen. Unfortunately, designing a good proposal distribution is generally hard.

We introduce a more general scheme which we call importance sampling over sets (ISS) where we randomly select (large) subsets of states (rather than individual samples) using a generalized notion of proposal distribution called set-proposal distribution. Like traditional importance sampling, we provide a way to re-weight the samples and obtain an unbiased estimator for the original high-dimensional integral of interest. Intuitively, the idea is that by considering a very large (potentially, even exponential in the dimensionality of the problem) number of samples, one can significantly reduce the variance. Unfortunately, simply enumerating the samples would take exponential space. We therefore consider specially structured set-proposal distributions such that the set of samples can be represented in an *implicit* and *compact* way. The second main obstacle to overcome is that it is no longer possible to do enumeration-based inference on the samples. We therefore propose an approximation based on the importance weight of the heaviest configuration in the sampled set. For many classes of probabilistic models, e.g. log-supermodular [6], we can compute these statistics efficiently, e.g. using graphcuts. Surprisingly, we can show some strong formal guarantees

for this approximation. In particular, we identify a natural link between our scheme and some recently introduced probabilistic inference schemes based on randomized hashing and optimization [7, 8]. By reformulating these prior results within our framework, we show that there exists set-proposal distributions that are in some sense *universal*—they are guaranteed to give accurate answers using a small number of samples no matter what the underlying probabilistic model is.

We improve the accuracy and efficiency of our approach by developing a class of *adaptive* set-proposal distributions that can be tailored to the specific target probabilistic model leveraging the samples we draw from the model. We show that this approach provides very accurate estimates for the partition function of undirected graphical models on a range of benchmark problems. Our method is also extremely scalable: we are able to estimate the partition function for models with *up to one million variables* in a matter of minutes. Finally, we develop a new maximum likelihood parameter learning scheme based on our probabilistic inference framework. Our technique is very different from standard gradient descent approaches, and resembles structured prediction schemes such as structured SVM learning. We empirically show the effectiveness of our technique on the standard MNIST handwritten digits dataset.

## 2 SETUP

Given an undirected graphical model with  $n$  binary variables, let  $\mathcal{X} = \{0, 1\}^n$  be the set of all possible configurations (variable assignments or possible states of the world). Define a weight function  $w : \mathcal{X} \rightarrow \mathbb{R}^+$  that assigns to each configuration  $x$  a score proportional to its probability  $p(x)$ :  $w(x) = \prod_{\alpha \in \mathcal{I}} \psi_{\alpha}(\{x\}_{\alpha})$ . The weight function is compactly represented as a product of factors or potentials. The *partition function* of the model  $Z$  is defined as  $Z = \sum_{x \in \mathcal{X}} w(x) = \sum_{x \in \mathcal{X}} \prod_{\alpha \in \mathcal{I}} \psi_{\alpha}(\{x\}_{\alpha})$ . It is a normalization constant used to guarantee that  $p(x) = w(x)/Z$  sums to one. Computing  $Z$  is typically intractable because it involves a sum over an exponential number of configurations, and is often the most challenging inference task for many families of graphical models. Computing  $Z$  is required for many inference and learning tasks, such as evaluating the likelihood of data for a given model, computing marginal probabilities, and comparing competing models of data [29, 17].

Given that probabilistic inference problems are intractable in the worst case [23], a number of approximate inference algorithms have been developed. There are two main families of algorithms: Monte Carlo sampling techniques and variational approximations. Variational methods are based on approximating the target distribution  $p$  using a family of tractable approximating distributions, and minimizing a notion of divergence. Sampling techniques are randomized

approaches where the key idea is to estimate statistics of interest by looking at a small number of representative states.

## 3 IMPORTANCE SAMPLING

The simplest (naive) approach is to sample  $x_1, \dots, x_M$  uniformly from  $\mathcal{X}$ , and estimate  $\hat{Z} = \frac{1}{M} \sum_{i=1}^M w(x_i) 2^n$ . This is an unbiased estimator of  $Z$  as  $\mathbb{E}[\hat{Z}] = \frac{1}{M} \sum_{i=1}^M \sum_{x \in \mathcal{X}} \frac{1}{2^n} 2^n w(x) = Z$ . The variance of this estimator can be very large since we are limited to a small number of samples  $M$ , while the number of possible configurations  $|\mathcal{X}|$  is exponential in  $n$ . The variance can be reduced using *importance sampling* (IS) techniques, i.e. sampling using a proposal distribution (which is closer to  $p(x)$ ) rather than uniformly [1, 15, 19]. Here,  $x_1, \dots, x_M$  are sampled from  $\mathcal{X}$  according to some proposal distribution  $q(x)$ , and weighted by their inverse likelihood,  $\hat{Z} = \frac{1}{M} \sum_{i=1}^M \frac{w(x_i)}{q(x_i)}$ . This is also an unbiased estimator for  $Z$ .

Unfortunately, it is usually the case that the closer the proposal distribution  $q$  is to the original intractable  $p(x)$ , the harder it gets to sample from it. Markov Chain Monte Carlo sampling is one of the leading approaches for sampling from arbitrary distributions [1, 15, 19]. The key idea is to draw proper representative samples from  $p(x)$  by setting up a Markov Chain over the entire state space which has to reach an equilibrium distribution. For many statistical models of interest, reaching the equilibrium distribution will require simulating the chain for a number of steps which is exponential in  $n$ . Unless there are special regularity conditions, if the random walk does not visit all the possible states it might miss some important parts. In practice, the approach will therefore only give approximate answers. There is generally little or no information on the quality of the approximation. In fact, the Markov Chain may get trapped in less relevant areas and completely miss important parts of the state space.

Most similar to our approach is Greedy Importance Sampling (GIS) [27], a reformulation of IS which achieves variance reduction by sampling blocks of variables from a proposal distribution and then searching for highly weighted regions in the target distribution. The blocks of points are non-overlapping and points within a block are ordered, allowing points in a block to be selected using a greedy search. This search increases the probability of blocks containing highly weighted points and outperforms naive methods which are unlikely to observe such points by chance. These blocks can be seen as a special-case of sets in the ISS technique, in which the sets are selected through search. Whereas GIS blocks are likely to contain highly weighted points due to explicit search, sets in ISS more generally contain highly weighted points by sampling any exponentially large subset of points and extracting statistics of interest. For example ISS allows the use of order-statistics (MAP/MPE estimation) which can often be com-

puted efficiently (e.g. using graphcuts, Viterbi), although the search method of GIS is another approach. The methods are orthogonal and future work can investigate incorporating explicit search or other techniques such as analytic marginalization within ISS to further reduce variance. Another key generalization of ISS is that sets of points can overlap, which grants additional freedom in selecting set-proposal distributions. For example when sets are defined by parity constraints, set-proposal distributions implement a strongly universal hash function, providing strong theoretical guarantees on the accuracy of the estimates.

## 4 IMPORTANCE SAMPLING OVER SETS

We propose a generalized importance sampling procedure, in which instead of randomly selecting a single configuration  $\mathbf{x}$  we randomly select a (large) subset of configurations  $S \subseteq \mathcal{X}$ . Let  $P(\mathcal{X})$  denote the power set of  $\mathcal{X} = \{0, 1\}^n$ , i.e. the set of all subsets of  $\mathcal{X}$ . We define a probability distribution  $q$  over  $P(\mathcal{X})$  as a **set-proposal distribution**. A set-proposal distribution induces the following function  $\gamma : \mathcal{X} \rightarrow [0, 1]$

$$\gamma(\mathbf{x}, q) = \sum_{S \in P(\mathcal{X})} \mathbf{1}(\mathbf{x} \in S)q(S) \quad (1)$$

Intuitively,  $\gamma(\mathbf{x}, q)$  is the probability of  $\mathbf{x}$  being contained in a set  $S$  sampled from  $q$ . We omit the dependency on  $q$  when the set-proposal distribution used is clear from the context. Standard proposal distributions used in Importance Sampling are a special case of set-proposal distributions, assigning zero probability to all subsets  $S \subseteq \mathcal{X}$  such that  $|S| \neq 1$ . The following results generalizes the standard importance sampling result to our more general case.

**Proposition 1.** *Let  $q$  be any set-proposal distribution such that  $w(\mathbf{x}) > 0$  implies  $\gamma(\mathbf{x}, q) > 0$ . Let  $S \sim q$  denote a random sample from the set-proposal distribution  $q$ . Then  $\sum_{\mathbf{x} \in S} \frac{w(\mathbf{x})}{\gamma(\mathbf{x}, q)}$  is an unbiased estimator for the partition function  $Z$ .*

*Proof.*

$$\begin{aligned} Z &= \sum_{\mathbf{x} \in \mathcal{X}} w(\mathbf{x}) = \sum_{\mathbf{x} \in \mathcal{X}} w(\mathbf{x}) \frac{\gamma(\mathbf{x})}{\gamma(\mathbf{x})} \\ &= \sum_{\mathbf{x} \in \mathcal{X}} \frac{w(\mathbf{x})}{\gamma(\mathbf{x})} \sum_{S \in P(\mathcal{X})} \mathbf{1}(\mathbf{x} \in S)q(S) \\ &= \sum_{S \in P(\mathcal{X})} q(S) \sum_{\mathbf{x} \in \mathcal{X}} \frac{w(\mathbf{x})}{\gamma(\mathbf{x})} \mathbf{1}(\mathbf{x} \in S) \\ &= \sum_{S \in P(\mathcal{X})} q(S) \sum_{\mathbf{x} \in S} \frac{w(\mathbf{x})}{\gamma(\mathbf{x})} = E_{S \sim q} \left[ \sum_{\mathbf{x} \in S} \frac{w(\mathbf{x})}{\gamma(\mathbf{x})} \right] \end{aligned}$$

□

Note that when the set-proposal distribution  $q$  is a standard proposal distribution (over singletons), one recovers the standard importance sampling result. There are three main aspects to consider for the practical usability of Proposition 1. We need to 1) sample a subset  $S$  from  $q$  efficiently, 2) evaluate the importance weight  $\gamma(\mathbf{x})$  tractably, 3) when  $S$  is (exponentially) large, represent  $S$  compactly and evaluate the summation. The first two considerations apply to traditional importance sampling as well. The third one is new. For example, if  $q$  deterministically chooses  $S = \mathcal{X}$ , then evaluating the estimator is just as hard as computing the partition function. As this extreme example suggests, the advantage is that by considering larger sets, one can significantly reduce the variance. The following corollary is very useful,

**Corollary 1.** *Let  $q$  be any set-proposal distribution such that  $w(\mathbf{x}) > 0$  implies  $\gamma(\mathbf{x}) > 0$ . Let  $S \sim q$  denote a random sample from the set-proposal distribution  $q$ . Then  $E_{S \sim q} \left[ \max_{\mathbf{x} \in S} \frac{w(\mathbf{x})}{\gamma(\mathbf{x})} \right]$  is a lower bound for the partition function  $Z$ .*

*Proof.* Since the weights are non-negative  $w(\mathbf{x}) \geq 0$ , it follows that  $\max_{\mathbf{x} \in S} \frac{w(\mathbf{x})}{\gamma(\mathbf{x})} \leq \sum_{\mathbf{x} \in S} \frac{w(\mathbf{x})}{\gamma(\mathbf{x})}$  and the claim follows from Proposition 1 by linearity of expectation. □

Notice that if  $q$  is a standard proposal distribution, i.e.  $q(S) = 0$  if  $|S| \neq 1$ , the estimators  $\sum_{\mathbf{x} \in S} \frac{w(\mathbf{x})}{\gamma(\mathbf{x})}$  and  $\max_{\mathbf{x} \in S} \frac{w(\mathbf{x})}{\gamma(\mathbf{x})}$  coincide. In general, the value of  $\max_{\mathbf{x} \in S} \frac{w(\mathbf{x})}{\gamma(\mathbf{x})}$  can be exponentially far from  $\sum_{\mathbf{x} \in S} \frac{w(\mathbf{x})}{\gamma(\mathbf{x})}$ , for example in the case of a constant (uniform) weight function  $w(\cdot)$ . The upside is that the max statistic, i.e. computing the mode of the distribution, is often more tractable. For example, there are numerous classes of probabilistic models, such as attractive Ising models, where one can find the mode of the distribution (MAP/MPE query) in polynomial time, while computing the partition function is NP-hard [11, 14].

### 4.1 EXAMPLES OF SET-PROPOSAL DISTRIBUTIONS

In both examples below, let  $m \leq n$ , let  $v_m(\mathbf{x}) = \{v_i(x_i), i = 1, \dots, m\}$  be a family of marginal distributions over individual variables. Let  $b_i$  be independent samples from  $v_i(x_i)$  for  $i = 1, \dots, m$ .

#### 4.1.1 Constraining Variables

We can define a set-proposal distribution  $q$  where to sample a set we define  $S = \{\mathbf{x} \in \mathcal{X} : x_i = b_i, \forall i \in \{1, \dots, m\}\}$ . Note that  $\gamma(\mathbf{x}) = \prod_{i=1}^m q(x_i) = \prod_{i=1}^m v_i(x_i)^{b_i} (1 - v_i(x_i))^{1-b_i}$ . The set can be represented compactly using  $m$  equations (equivalently, additional factors to be added

to the graphical model that clamp some variables to certain values). Intuitively, this approach samples a set  $S$  where  $|S| = 2^{n-m}$  by constraining or "clamping" variables  $x_1, \dots, x_m$  to fixed binary values.

### 4.1.2 Parity Constraints

As a second example of a set-proposal distribution, let  $A \in \{0, 1\}^{m \times n}$  be a binary matrix with rows  $a_i$ . We define a set-proposal distribution  $q$  according to the following generative process. To sample a set  $S$  from  $q$ , we define  $S = \{\mathbf{x} \in \mathcal{X} : a_i \mathbf{x} = b_i \pmod{2}, \forall i \in \{1, \dots, m\}\}$ . It can be seen that given any  $\mathbf{x} \in \mathcal{X}$ , the probability that  $\mathbf{x}$  belongs to a randomly chosen  $S \sim q$  is again  $\gamma(\mathbf{x}) = \prod_{i=1}^m q(x_i)$ . This is the probability that  $\mathbf{x}$  satisfies  $m$  parity equations with randomly chosen right-hand side coefficients. The set can be represented compactly using  $m$  linear equations modulo 2. Parity constraints can be represented compactly as a product of factors, using a linear number of extra variables [7].

## 5 MULTIPLE PROPOSAL DISTRIBUTIONS

As noted earlier, the lower bound obtained with Corollary 1 given by  $L(S, q) = \max_{\mathbf{x} \in S} \frac{w(\mathbf{x})}{\gamma(\mathbf{x}, q)}$  might be loose compared to  $A(S, q) = \sum_{\mathbf{x} \in S} \frac{w(\mathbf{x})}{\gamma(\mathbf{x}, q)}$ , which is an unbiased estimator of  $Z$  by Proposition 1. Here we are making explicit the dependence of  $\gamma(\mathbf{x}, q)$  on the set-proposal distribution  $q$ . Intuitively, this approximation is accurate when the weight distribution over  $S$  is peaked, i.e. the mode is a good approximation of the "total area". On the other hand, the lower bound is loose when there are "many" configurations in  $S$  that have a weight comparable to the one of the heaviest assignment. If that is the case, it is intuitively possible to randomly subsample the set  $S$  and obtain a smaller set  $S' \subseteq S$  such that  $\max_{\mathbf{x} \in S} w(\mathbf{x}) \approx \max_{\mathbf{x} \in S'} w(\mathbf{x})$ . Since  $|S'| < |S|$ , the gap between the approximation introduced by considering only the mode of the weight distribution on  $S'$  yields a smaller error. This suggests the use of another set-proposal distribution  $q'$  that is more likely to propose smaller sets  $S'$  compared to  $q$ . Because we do not know a priori if the bound is tight or not, this discussion motivates a more general scheme that relies on multiple set-proposal distributions. By letting the typical size of a sampled set change, e.g. from 1 to  $|\mathcal{X}| = 2^n$ , we can sample from a wide variety of configurations and accurately predict  $\log Z$  for distributions which are peaked to various degrees.

**Proposition 2.** Let  $\mathcal{Q} = (q_1, \dots, q_k)$  be a family of set-proposal distributions. Let  $S_1 \sim q_1, S_2 \sim q_2, \dots, S_k \sim q_k$ . Suppose that for all  $i$ ,  $w(\mathbf{x}) > 0$  implies  $\gamma(\mathbf{x}, q_i) > 0$ . Then  $\frac{1}{k} \sum_{i=1}^k \sum_{\mathbf{x} \in S_i} \frac{w(\mathbf{x})}{\gamma(\mathbf{x}, q_i)}$  is an unbiased estimator for the partition function  $Z$ .

*Proof.*

$$E_{S_1 \sim q_1, \dots, S_k \sim q_k} \left[ \frac{1}{k} \sum_{i=1}^k \sum_{\mathbf{x} \in S_i} \frac{w(\mathbf{x})}{\gamma(\mathbf{x}, q_i)} \right] = \frac{1}{k} \sum_{i=1}^k E_{S_i \sim q_i} \left[ \sum_{\mathbf{x} \in S_i} \frac{w(\mathbf{x})}{\gamma(\mathbf{x}, q_i)} \right] = Z$$

where the last step follows from Proposition 1.  $\square$

The following corollary follows, and is implemented by Algorithm 1 with input  $T = 1$ .

**Corollary 2.** Let  $\mathcal{Q} = (q_1, \dots, q_k)$  be a family of set-proposal distributions. Let  $S_1 \sim q_1, S_2 \sim q_2, \dots, S_k \sim q_k$ . Suppose that for all  $i$ ,  $w(\mathbf{x}) > 0$  implies  $\gamma(\mathbf{x}, q_i) > 0$ . Then  $\frac{1}{k} \sum_{i=1}^k \max_{\mathbf{x} \in S_i} \frac{w(\mathbf{x})}{\gamma(\mathbf{x}, q_i)}$  is in expectation a lower bound for the partition function  $Z$ .

*Proof.* Follows immediately from Corollary 1.  $\square$

**Input** :  $w : \mathcal{X} \rightarrow \mathbb{R}^+, T, k, q_i$  for  $i = 1, \dots, k$

**Output**: Estimate of  $\log Z = \log \sum_{\mathbf{x} \in \mathcal{X}} w(\mathbf{x})$

```

1 for  $i = 1, \dots, k$  do
2   Sample  $S_i^1 \dots S_i^T$  according to  $q_i$ 
3   for  $t = 1, \dots, T$  do
4      $\mathbf{m}_i^t = \max_{\mathbf{x} \in S_i^t} \frac{w(\mathbf{x})}{\gamma(\mathbf{x}, q_i)}$ 
5   end
6    $M_i \leftarrow \text{Median}(\mathbf{m}_i^1 \dots \mathbf{m}_i^T)$ 
7 end
8 Return  $\frac{1}{k} \sum_{i=1}^k M_i$ 

```

**Algorithm 1:** Set importance sampling

If we now let the typical size of a sampled set change, e.g. from  $|S| = 1$  to  $|S| = |\mathcal{X}| = 2^n$ , the magnitude of the various  $\gamma(\mathbf{x}, q_i)$  varies exponentially. Therefore it is often the case that many terms in the sum  $\sum_{i=1}^k \max_{\mathbf{x} \in S_i} \frac{w(\mathbf{x})}{\gamma(\mathbf{x}, q_i)}$  will not contribute significantly to the overall estimate of  $Z$ . In practice, a sufficiently accurate lower bound is obtained by  $T$  samples of only the highest weighted sets,

$$\max_{i=1}^k \max_{\mathbf{x} \in S_i} \frac{w(\mathbf{x})}{\gamma(\mathbf{x}, q_i)} \quad (2)$$

The main advantage of this approach is computational, as we have now reduced the inference procedure to a *single* optimization problem. The following result shows that this still provides a valid lower bound:

**Corollary 3.** Let  $\mathcal{Q} = (q_1, \dots, q_k)$  be a family of set-proposal distributions. Suppose that for all  $i$ ,  $w(\mathbf{x}) > 0$  implies  $\gamma(\mathbf{x}, q_i) > 0$ . Let  $S_i^1, \dots, S_i^T \sim q_i$  be i.i.d samples from the  $i$ -th set-proposal distribution  $q_i$ . Then  $\max_{i=1}^k \text{Median} \left( \max_{\mathbf{x} \in S_i^1} \frac{w(\mathbf{x})}{\gamma(\mathbf{x}, q_i)}, \dots, \max_{\mathbf{x} \in S_i^T} \frac{w(\mathbf{x})}{\gamma(\mathbf{x}, q_i)} \right)$  is with high probability an approximate lower bound for the partition function  $Z$ .



*Proof.* For every  $i$ , let us denote  $L(S_i) = \max_{\mathbf{x} \in S_i} \frac{w(\mathbf{x})}{\gamma(\mathbf{x}, q_i)}$ . Then by Proposition 1,  $E_{S_i \sim q_i} [L(S_i)] \leq Z$ . Therefore by Markov's inequality,  $P \left[ \max_{\mathbf{x} \in S_i} \frac{w(\mathbf{x})}{\gamma(\mathbf{x}, q_i)} \geq 4Z \right] \leq \frac{1}{4}$ . Let  $S_i^1, \dots, S_i^T$  be samples from  $q_i$ . It follows from Chernoff's inequality that

$$P[\text{Median}(L(S_i^1), \dots, L(S_i^T)) \geq 4Z] \leq \exp\left(-\frac{T}{24}\right)$$

therefore from the union bound

$$P[\cup_{i=1}^k \text{Median}(L(S_i^1), \dots, L(S_i^T)) \geq 4Z] \leq k \exp\left(-\frac{T}{24}\right)$$

Therefore

$$P[\max_{i=1}^k \text{Median}(L(S_i^1), \dots, L(S_i^T)) \leq 4Z] \geq 1 - k \exp\left(-\frac{T}{24}\right)$$

□

Corollary 3 is implemented in Algorithm 1 with line 8 changed to return the maximum  $M_i$  instead of the mean.

To make the procedure of Corollary 3 clear, consider again a family of set-proposal distributions  $q_i$  constructed by constraining variables as in the example of section 4.1.1. This can be implemented in Algorithm 1 by setting  $k = n + 1$  and selecting  $q_i$  to constrain the first  $i - 1$  variables: The outer-loop (line 1) searches for the  $q_i$  which contributes most towards the estimate of  $\log Z$  by, in each iteration, enforcing  $i$  "hard" variable constraints which limit the set of possible configurations by defining sets  $S_i^t$  where  $|S_i^t| = 2^{n+1-i}$ . Notice under this setup that if the maximum iteration is  $i = 1$  (no variables constrained,  $|S| = |\mathcal{X}| = 2^n$ ), then this is equivalent to approximating  $\log Z$  by the MAP/MPE configuration. Conversely, if the maximum is at  $i = n + 1$ , then this is equivalent to naive importance sampling based on proposal distribution  $q_{n+1}$  (all variables constrained,  $|S| = 1$ ). If the heaviest weighted set is not one of these two special cases, then the set importance sampling method will produce a more accurate estimate of  $\log Z$ . Also note that since empirically most of the iterations do not contribute significantly to the overall estimate of  $\log Z$ , the outer loop in Algorithm 1 over all  $n$  variables can search with a larger granularity or logarithmically for the heaviest  $i$ . In practice, fixing the number of iterations in the outer loop to 10 (sampling sets with a granularity of  $\frac{n}{10}$  constrained variables) is both accurate and fast to run. In fact, this can be taken a step further by skipping the loop altogether and searching for the heaviest weighted set as a *single* optimization problem: rather than a loop which incrementally adds "hard" variable constraints, we can add all the variable constraints at once as "soft" constraints which an optimization oracle may choose to satisfy. The reward for satisfying these constraints matches the scaling  $\frac{1}{\gamma(\mathbf{x}, q_i)}$ . Formulating the estimate of  $\log Z$  as a single optimization problem is useful for learning, see section 7.

## 5.1 RELATIONSHIP WITH RANDOMIZED HASHING

The advantage of using multiple proposal distributions is that one might be able to reduce the variance of the estimator, in accordance with the intuitive motivation presented earlier. In fact, it can be shown that there exists set-proposal distributions (based on universal hash functions) such that a polynomial number of samples is sufficient to obtain concentration of the estimate around the mean. The surprising result is that these proposal distributions are "universal", in that they are guaranteed to give accurate estimates (constant factor approximations) for *any* weight function  $w(\cdot)$ , i.e., any underlying graphical model.

Let  $\mathcal{S} \subseteq P(\mathcal{X})$  be a family of sets defined as  $\mathcal{S} = \{\{\mathbf{x} \in \mathcal{X} : \mathbf{A}\mathbf{x} = \mathbf{b} \bmod 2\}, A \in \{0, 1\}^{i \times n}, b \in \{0, 1\}^i\}$ . Let  $q_i$  be a set-proposal distribution where to sample from  $q_i$  we randomly choose each entry of  $A, b$  uniformly at random (independently). Then it can be shown that  $\gamma(\mathbf{x}) = 2^{-i}$ . For a state space  $\mathcal{X} = \{0, 1\}^n$ , let us consider a family of  $n$  proposal distributions  $\mathcal{Q}_P = (q_0, \dots, q_n)$ . These set-proposal distributions can be interpreted as implementing a strongly universal hash function, where each element  $\mathbf{x} \in \mathcal{X}$  is sampled by the  $i$ -th proposal distribution  $q_i$  with probability  $2^{-i}$ , and elements are sampled pairwise independently [9, 3, 12, 7, 8, 13]. As noted above, we can sample from  $q_i$  tractably, and represent sets  $S \in \mathcal{S}$  in a compact way. Theorem 1 from [8] implies the following remarkable result

**Corollary 4.** *Let  $\mathcal{Q}_P = (q_0, \dots, q_n)$  be defined as above. Let  $L(S_i) = \max_{\mathbf{x} \in S_i} \frac{w(\mathbf{x})}{\gamma(\mathbf{x}, q_i)}$  for every  $i$ . Then for any weight function  $w(\cdot)$  and  $1 > \delta > 0$ ,  $\sum_{i=1}^n \text{Median}(L(S_i^1), \dots, L(S_i^T))$  is with probability at least  $1 - \delta$  a constant factor approximation of  $Z$  when  $T = \Theta(n \ln n / \delta)$ .*

Reinterpreted in our set-proposal distribution framework, Corollary 4 is important because it shows that there exists a family of universal set-proposal distributions that are guaranteed to work well for any underlying target probability distribution  $p$ .

Although sets  $S \in \mathcal{S}$  defined by parity constraints can be represented compactly, the resulting optimization problems  $\max_{\mathbf{x} \in S_i} \frac{w(\mathbf{x})}{\gamma(\mathbf{x}, q_i)}$  are generally difficult to solve, even when  $w(\cdot)$  can be tractably optimized, i.e.,  $\max_{\mathbf{x} \in \mathcal{X}} w(\mathbf{x})$  can be solved efficiently. Rather than take a worst-case approach and consider a proposal distribution that is guaranteed to work for any weight function  $w$  as in [9, 3, 12, 7, 8, 13], in this paper we consider a more general class of set-proposal distributions. In particular, we construct proposal distributions that are tailored to particular probabilistic models (and weight function  $w$ ). The main advantage of this approach is that we can leverage the structure of the original problem, and the corresponding optimization problems

will be easier to solve, leading to massive improvements in scalability. This is similar in spirit to traditional importance sampling, where one typically uses some prior knowledge on the underlying probabilistic model to construct good proposal distributions.

## 6 ADAPTIVE SET IMPORTANCE SAMPLING SCHEME

Similarly to standard adaptive importance sampling schemes (e.g. [22]), we propose an adaptive procedure where set-proposal distributions are adapted based on the samples collected. This is an enhancement of Algorithm 1 in that its iterative procedure can be exploited to adaptively improve the input set-proposal distributions.

Recall from section 4.1.1 that a set-proposal distribution can be defined in which sets  $S_i$  are sampled by constraining variables  $x_1, \dots, x_i$ . For a fixed  $i$  there are  $2^i$  such sets  $S_i$  (where  $|S_i| = 2^{n-i}$ ), and as in the previous section for each  $i$  we sample  $T$  such sets  $S_i^t$ . Next, let  $\mathbf{e}_i^t = \arg \max_{\mathbf{x} \in S_i^t} \frac{w(\mathbf{x})}{\gamma(\mathbf{x}, q_i)}$ . We define empirical marginal distributions  $\hat{v}_{i+1}(\mathbf{x})$  based on the fraction of samples  $\mathbf{e}_i^t$  that have variables  $x_1$  to  $x_{i+1}$  set to one (with Laplace smoothing). Intuitively, the adaptive set importance algorithm performs the same iteration as Algorithm 1, sampling sets by incrementally constraining variables  $x_1$  to  $x_i$ , except that it interpolates the *input* proposal distribution with the empirical marginal distributions from the previous iteration to define a new set-proposal distribution for the current iteration. The full details of the algorithm are shown in Algorithm 2. During each iteration, as in Algorithm 1, generate  $T$  sets  $S_i^1 \dots S_i^T$  where each represents a set of configurations  $\mathbf{x}$  in which the first  $i$  binary variables are "clamped" to 0 or 1. The solutions  $\mathbf{e}_i^t = \arg \max_{\mathbf{x} \in S_i^t} \frac{w(\mathbf{x})}{\gamma(\mathbf{x}, q_i)}$  produce  $T$  samples which define the empirical marginal distribution  $\hat{v}_{i+1}(\mathbf{x})$ , and this empirical marginal distribution is used to sample sets in iteration  $i+1$ . For the first iteration, the variable  $x_1$  is sampled according to any proposal distribution (uniformly by default). Note also that when obtaining the MAP configuration of many variables is intractable, the iteration can begin with any number of variables constrained according to any proposal distribution (not just  $x_1$ ). ISS is still guaranteed to perform at least as well (and often much better) as IS by selecting subsets small enough that calculating the mode by brute force enumeration is tractable.

## 7 LEARNING

Because set importance sampling provides fast and scalable partition function estimates, it can be used for learning. We consider the standard problem of maximum likelihood learning of the parameters of an undirected graphical model. Given samples  $\mathbf{x}_1, \dots, \mathbf{x}_M$  from a parameterized probability distribution  $p_\theta(\mathbf{x}) = \frac{1}{Z(\theta)} \exp(\theta \phi(\mathbf{x}))$ ,

**Input** :  $w : \mathcal{X} \rightarrow \mathbb{R}^+, T$

**Output**: Estimate of  $\log Z = \log \sum_{\mathbf{x} \in \mathcal{X}} w(\mathbf{x})$

- 1  $M_0 \leftarrow \arg \max_{\mathbf{x} \in \mathcal{X}} w(\mathbf{x})$
- 2 Define  $\hat{v}_1(\mathbf{x})$  as uniform marginal over  $x_1$
- 3 **for**  $i = 1, \dots, n$  **do**
- 4     Sample  $S_i^1 \dots S_i^T$  using marginals  $\hat{v}_i(\mathbf{x})$  as in 4.1.1
- 5     **for**  $t = 1, \dots, T$  **do**
- 6          $\mathbf{m}_i^t, \mathbf{e}_i^t = \max, \arg \max_{\mathbf{x} \in S_i^t} \frac{w(\mathbf{x})}{\gamma(\mathbf{x}, q_i)}$
- 7     **end**
- 8      $M_i \leftarrow \text{Median}(\mathbf{m}_i^1 \dots \mathbf{m}_i^T)$
- 9     Compute  $\hat{v}_{i+1}(\mathbf{x})$  based on  $\mathbf{e}_i^1 \dots \mathbf{e}_i^T$  (with Laplace smoothing). This is the fraction of argmax results  $\mathbf{e}_i^t$  with  $x_j = 1$ , for  $j \in \{1, \dots, i+1\}$ .
- 10 **end**
- 11 Return  $\frac{1}{n+1} \sum_{i=0}^n M_i$

**Algorithm 2:** Adaptive set importance sampling

find maximum likelihood estimate of the parameters

$$\max_{\theta} \sum_{i=1}^M \log p_{\theta}(x_i) \quad (3)$$

which can be equivalently written as  $\max_{\theta} \theta \frac{1}{M} \sum_{i=1}^M \phi(x_i) - \log Z(\theta)$ . It is well known that solving this parameter learning problem is very challenging because it requires inference to evaluate the partition function (or its gradient). In this section we show how our importance sampling scheme can be used to approximate the value of the partition function, leading to a new learning algorithm. The algorithm we obtain is similar to structured prediction learning algorithms and cutting plane techniques [30, 28, 26], used for example in training structured support vector machines. A key difference is that our approach is used to (approximately) optimize the likelihood (in a generative setting), rather than minimizing a loss function in a discriminative setting.

### 7.1 LEARNING ALGORITHM

Structured support vector machines (SSVM) [30] and other structured prediction learning methods [18, 5] are trained by solving a convex optimization problem in which the number of constraints is exponential. The problems can be solved tractably by iteratively constructing a sufficient subset of constraints and employing an optimization oracle to find the next constraint to include. In this way the subset of the constraints is enlarged iteratively and provides a lower bound on the optimization objective.

The learning approach based on set importance sampling is similar, but using the set importance sampling technique as an optimization oracle. Beginning from the logarithm of equation (3),  $\max_{\theta} \theta \frac{1}{M} \sum_{i=1}^M \phi(\mathbf{x}_i) - \log Z(\theta)$ , we can introduce a variable  $\alpha$  which takes the place of  $\log Z$  and cast the optimization as follows

$$\begin{aligned} & \underset{\theta, \alpha}{\text{maximize}} && \theta \frac{1}{M} \sum_{i=1}^M \phi(\mathbf{x}_i) - \alpha \\ & \text{subject to} && \alpha \geq \log Z(\theta) \end{aligned}$$

We then express  $Z(\theta)$  using the approximation given by equation (2) as

$$\alpha \geq \log Z(\theta) \geq \max_{i=1}^k \max_{\mathbf{x} \in \mathcal{S}_i} \theta \phi(\mathbf{x}) - \log \gamma(\mathbf{x}, q_i) \quad (4)$$

Note this is an exponentially large set of *linear* constraints in  $\theta$  and  $\alpha$ , and therefore corresponds to a linear program (LP). Because the number of constraints is exponential in the number of variables, as in structured learning we consider only a subset  $C$  of constrained configurations  $\mathbf{x}$ . The reduced LP becomes,

$$\begin{aligned} & \underset{\theta}{\text{maximize}} && \theta \frac{1}{M} \sum_{i=1}^M \phi(\mathbf{x}_i) - \alpha \\ & \text{subject to} && \alpha \geq \theta \phi(\mathbf{x}) + \beta(\mathbf{x}) \quad \forall \mathbf{x} \in C \end{aligned} \quad (5)$$

where  $\beta(\mathbf{x}) = \max_{i | \mathbf{x} \in \mathcal{S}_i} (-\log \gamma(\mathbf{x}, q_i))$  and intuitively is the maximum importance weight for a given  $\mathbf{x}$  under all set-proposal distributions  $q_i$  (see Appendix A).  $C$  is initially set to the training data set  $\{\mathbf{x}_1, \dots, \mathbf{x}_M\}$ , and is enlarged during each learning iteration by searching for the most violated constraint, i.e.  $\max_x \theta \phi(\mathbf{x}) + \beta(\mathbf{x})$ .

The full learning procedure is described in Algorithm 3. The input to the algorithm are the  $M$  training examples, the vector of sufficient statistics  $\phi$  and the (optional) choice of set-proposal distributions  $q_1 \dots q_k$  (for example uniform, based on the training examples, or adaptive if no  $q_i$  are provided). Each iteration of learning begins by finding the optimal weights for the LP in equation (5). Following the solution of the LP, we obtain the pair  $(\theta_i, \alpha_i)$ . Then, using these learned weights  $\theta_i$ , importance sampling over sets is used to approximate  $\log Z$  for various set-proposal distributions and importance weights, and each of these samples (modes) is added to constraint set  $C$ . Note that Algorithm 3 takes advantage of an optimization oracle to solve the LP in equation (5). Another optimization oracle is used within our importance sampling over sets procedure to estimate  $\log Z$  (by optimizing equation (4) using the current parameter estimate  $\theta_i$ ). Intuitively, the value of the partition function is not just approximated using the MAP assignment, but thanks to the importance weights, we are able to obtain better estimates. For example, if using the current weight vector  $\theta_i$  the distribution is close to uniform, an approximation based on the MAP assignment would be poor, but we can still get good approximation to the partition function thanks to the importance weights.

Because at each iteration a (convex) linear program is solved to obtain the weights  $\theta$ , at each iteration the weights obtained for the constraints  $C$  are guaranteed to be globally optimal for the approximate likelihood objective. This

**Input** :  $\mathbf{x}_m, m = 1, \dots, M, \phi(\mathbf{x}), T, q_i, i = 1, \dots, k$   
**Output**: Learned weight parameters  $\theta$

```

1 converged ← False
2 i ← 0
3 C ← { $\mathbf{x}_m, m = 1, \dots, M$ }
4 while not converged do
5   i ← i + 1
6    $\theta_i, \alpha_i =$  solve LP (5) subject to C
7   for t = 1, ..., T do
8     |  $\mathbf{x}_{i,t}^*, \log Z_{est,t} =$  Run ISS with  $\theta_i, \{q_1, \dots, q_k\}$ 
9   end
10   $\log Z_{est} \leftarrow$  median  $\log Z_{est,t}$ 
11  if  $\alpha_i \geq \log Z_{est}$  then
12    | converged ← True
13  else
14    | C ← C  $\cup \{\mathbf{x}_{i,t}^*, t = 1, \dots, T\}$ 
15  end
16 end
17 Return  $\theta_i$ 

```

**Algorithm 3:** Iterative learning algorithm

is in contrast to gradient-based learning algorithms. Moreover, as more constraints are added to  $C$  at each iteration,  $C$  approaches  $\mathcal{X}$  and the LP objective is guaranteed to decrease monotonically towards the optimal approximate log likelihood of the training data.

## 8 EXPERIMENTAL RESULTS

### 8.1 PARTITION FUNCTION

One application of importance sampling over sets is for problems in which computing  $\max_{\mathbf{x}} w(\mathbf{x})$  is tractable, but  $\sum_{\mathbf{x}} w(\mathbf{x})$  is intractable. An example are functions which are log-supermodular. For such problems we can leverage fast optimization (for example using graph cuts), as long as  $w(\mathbf{x})/\gamma(\mathbf{x})$  stays tractable. For example, it is sufficient that  $\log(1/\gamma(\mathbf{x}))$  is supermodular.

We evaluated importance sampling over sets (ISS) against standard inference algorithms: junction tree (EXACT), belief propagation (BP), mean-field (MF), the MAP configuration (MAP), and naive importance sampling (IS). For junction tree, belief propagation and mean-field, the libDAI library was used [20].

First, we evaluated importance sampling over sets for functions which are not log-supermodular, to demonstrate the effectiveness of the method on general models. Table 1 shows estimates of the log partition function of the Ising Models from the 2011 Probabilistic Inference Challenge (PIC2011)<sup>1</sup>. These models have “mixed” interactions and therefore are not log-supermodular. For general functions

<sup>1</sup> [www.cs.huji.ac.il/project/PASCAL/showNet.php](http://www.cs.huji.ac.il/project/PASCAL/showNet.php)

which are not log-supermodular, each  $\arg \max$  in ISS was solved as an integer quadratic program (IQP) in CPLEX, with a search granularity of  $\frac{n}{10}$  to find the heaviest sets as discussed in section 5. The log partition function estimates in Table 1 use a uniform proposal distribution to show that even in the absence of a proposal distribution the ISS method performs better than both naive importance sampling and the MAP configuration (which are both special-cases of ISS).

Our second experiment validated ISS for functions which are log-supermodular, and thus for which ISS can be run in polynomial time. Fig. 1 evaluates the importance sampling over sets method on attractive (log-supermodular) Ising models with varying coupling strengths. Models have a field factor of 2.0, although we observed that a range of field factors gave almost identical results. For these log-supermodular potentials, optimization was performed with graph cuts using the OpenGM [2] library. For these experiments, importance sampling algorithms (IS and ISS) use the adaptive importance sampling scheme as a proposal distribution, where first ISS was run and the final empirical marginals in iteration  $n$  were also used for IS.

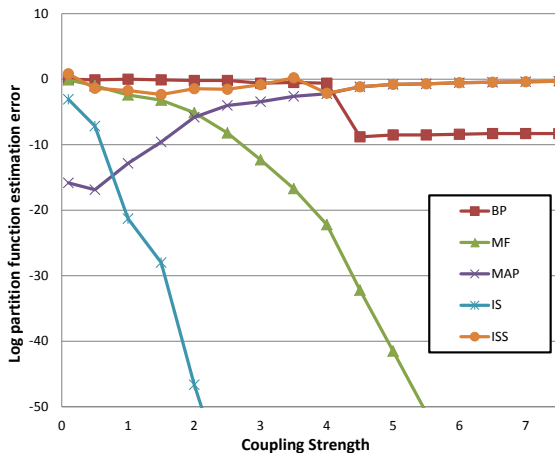


Figure 1: Log partition function error of various inference algorithms for 10x10 Ising grids with attractive (log-supermodular) interactions, a field factor of 2.0, and various coupling strengths. Importance sampling (IS) and importance sampling over sets (ISS) use adaptive set importance sampling.

Across a range of attractive Ising models, the set importance sampling technique provides very accurate estimates of the log partition function. Moreover, due to the log-supermodularity of the potentials, the ISS technique scales to much larger models, providing accurate estimates in polynomial time while other algorithms fail to converge. Tables 2 and 3 show the log partition function estimates and run-times for various Ising grid sizes, ranging from 10x10

Table 2: Log partition function estimates for various Ising model sizes. "-" indicates that no solution was obtained after 10 minutes. As in Table 1, ISS estimates are between MF (which tends to under-estimate) and BP (which tends to over-estimate)

|      | BP     | MF     | MAP    | IS     | ISS    |
|------|--------|--------|--------|--------|--------|
| 10   | 207.6  | 202.7  | 202.0  | 161.2  | 206.4  |
| 20   | 840.3  | 817.7  | 825.3  | 593.5  | 832.3  |
| 50   | 5195   | 4940   | 5110   | 3704   | 5125   |
| 100  | 20930  | 19910  | 20679  | 14670  | 20690  |
| 300  | 1.91E5 | 1.82E5 | 1.88E5 | 1.35E5 | 1.88E5 |
| 1000 | 2.11E6 | -      | 2.09E6 | 1.48E6 | 2.09E6 |

Table 3: Time (in seconds) to estimate logZ for Ising model sizes. "-" indicates that the algorithm did not converge within 10 minutes.

|      | EXACT | BP  | MF  | MAP | IS | ISS |
|------|-------|-----|-----|-----|----|-----|
| 10   | 1     | 1   | 1   | 1   | 1  | 1   |
| 20   | -     | 1   | 1   | 1   | 1  | 1   |
| 50   | -     | 5   | 8   | 1   | 1  | 5   |
| 100  | -     | 15  | 112 | 1   | 1  | 3   |
| 300  | -     | 119 | -   | 8   | 1  | 27  |
| 1000 | -     | -   | -   | 105 | 15 | 300 |

to 1000x1000. Notably, for the 300x300 models mean-field did not converge, but was still run for 10 minutes to give a solution. Similarly for 1000x1000 models, belief propagation did not converge but gave a solution after 10 minutes. For 1000x1000 models mean-field did not complete a single iteration within 10 minutes.

Finally, we extend the evaluation beyond Ising models by analyzing restricted Boltzmann machines (RBMs). Table 4 shows log partition function estimates for the largest RBM in [25] (784 visible units, 500 hidden units, trained on the MNIST dataset). AIS is the Annealed Importance Sampling technique described in that work. BP failed to converge. MF converged quickly but was less accurate than AIS. The quick convergence of mean-field was also noted by [24]. AIS was run in two modes, "no data" which estimated logZ from the model alone, and "data" which additionally used the training data to initialize the algorithm. In a similar spirit, due to the quick convergence of MF, and further demonstrating the flexibility of ISS to use any choice of proposal distribution, we ran mean-field to obtain marginals and used these as the proposal distribution for both IS and ISS. By leveraging MF as a proposal distribution ISS matches the accuracy of AIS with data. The ISS approach is valid even when no data is available.

## 8.2 LEARNING

In this final section we present preliminary analysis and empirical justification for the learning algorithm. We gen-

Table 1: Comparison of methods estimating the log partition function for Ising Models. The Importance Sampling (IS) and Importance Sampling over Sets (ISS) methods uses a uniform proposal distribution run over 5 random seeds, with the median presented. Shown in brackets next to ISS is the median number of constrained variables in the heaviest weighted set. The best estimate for each model is shown in bold.

|                    | <b>EXACT</b> | <b>BP</b>    | <b>MF</b> | <b>MAP</b> | <b>IS</b> | <b>ISS (c)</b>    |
|--------------------|--------------|--------------|-----------|------------|-----------|-------------------|
| grid10x10.f10      | 697.9        | 738.2        | 601.6     | 695.8      | 20.4      | <b>697.8</b> (3)  |
| grid10x10.f10.wrap | 767.5        | 837          | 695.4     | 766.5      | 65.85     | <b>767.9</b> (2)  |
| grid10x10.f15.wrap | 1146         | 1247         | 1036      | 1145.2     | 65.2      | <b>1146.6</b> (2) |
| grid10x10.f5.wrap  | 390.1        | 419.7        | 355.1     | 387.8      | 66.4      | <b>389.2</b> (2)  |
| grid20x20.f10      | 3021         | 3234         | 2592      | 3015.7     | 299.1     | <b>3017.1</b> (2) |
| grid20x20.f15      | 4520         | 4756         | 3947      | 4517.3     | 309.3     | <b>4518.7</b> (2) |
| grid20x20.f2       | 671.7        | <b>677.9</b> | 621.6     | 635.7      | 282.9     | 637.8 (21)        |
| grid20x20.f5       | 1531         | 1674         | 1391      | 1521.6     | 289.0     | <b>1522.4</b> (1) |

Table 4: Log partition function estimates for a restricted Boltzmann machine (RBM) trained on the MNIST dataset. Annealed importance sampling (AIS) was run with and without MNIST data for initialization. BP did not converge. IS and ISS were initialized with mean-field marginals as a proposal distribution and require no data.

| <b>Algorithm</b> | <b>logZ</b> |
|------------------|-------------|
| AIS (no data)    | 446.2       |
| AIS (data)       | 451.1       |
| BP               | -           |
| MF               | 437.5       |
| MAP              | 71.6        |
| IS               | 447.2       |
| ISS              | 450.2       |

eratively trained a naive Bayes model represented as an MRF on the MNIST handwritten digit dataset (size 28x28 images) and observed the algorithm’s capability to learn weights which accurately modeled the data. The learned model contained 794 variables and 7840 parameters. Examples were used as described in Algorithm 3. Fig. 2 shows a visualization of the learned weights as training progresses. The top image in Fig. 2 shows weights after 5 iterations of training, while the bottom image shows weights after 1000 iterations. Early in training the model captures with high confidence the most common patterns in the digits, but also noise. As training progresses, the model learns to generalize and differentiate between random noise and statistical variations in the data. Adaptive ISS was used as a proposal distribution, and similar results were obtained using marginals defined by the training data.

A straightforward extension of this work is extending the learning to latent variable models such as restricted Boltzmann machines, which the cutting-plane technique may be well-suited to given the accuracy of ISS in estimating RBM partition functions. We leave learning larger and more complex models to future work as the current contribution focuses on the importance sampling over sets technique.

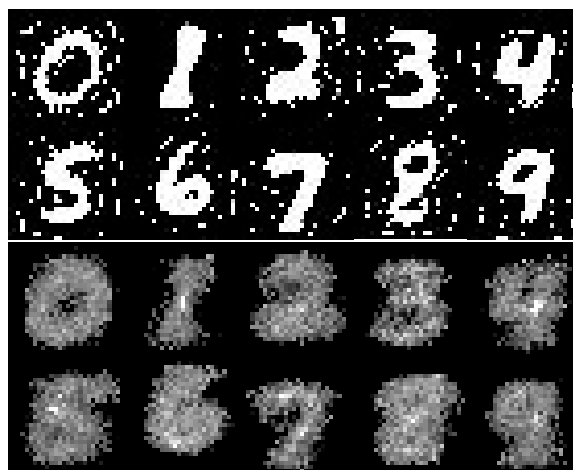


Figure 2: Visualization of learned weights of an undirected naive Bayes model trained generatively on the MNIST dataset. The top image shows weights after 5 iterations of training while the bottom image shows weights after 1000 iterations. Large positive weights are shown in white and large negative weights are shown in black.

## 9 CONCLUSIONS

We introduced a novel probabilistic inference algorithm called importance sampling over sets, based on randomly selecting (exponentially large) subsets of states rather than individual ones as in traditional importance sampling. By solving MAP inference queries over the sampled sets we obtain estimates of the partition function of undirected graphical models. This idea was incorporated into a novel maximum likelihood learning algorithm where the optimization oracle was used to obtain cutting planes. We demonstrated empirically that our scheme provides accurate answers on a range of benchmark instances and scales to very large problems with up to a million variables.

## References

- [1] Christophe Andrieu, Nando de Freitas, Arnaud Doucet, and Michael I Jordan. An introduction to MCMC for machine learning. *Machine learning*, 50(1-2):5–43, 2003.
- [2] T. Beier B. Andres and J. H. Kappes. OpenGM: A C++ library for discrete graphical models. *ArXiv e-prints*, 2012.
- [3] S. Chakraborty, K. Meel, and M. Vardi. A scalable and nearly uniform generator of SAT witnesses. In *Proc. of the 25th International Conference on Computer Aided Verification (CAV)*, 2013.
- [4] Barry A Cipra. The best of the 20th century: editors name top 10 algorithms. *SIAM news*, 33(4):1–2, 2000.
- [5] Michael Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 1–8. Association for Computational Linguistics, 2002.
- [6] Josip Djolonga and Andreas Krause. From MAP to marginals: Variational inference in Bayesian submodular models. In *Neural Information Processing Systems (NIPS)*, 2014.
- [7] Stefano Ermon, Carla P. Gomes, Ashish Sabharwal, and Bart Selman. Optimization with parity constraints: From binary codes to discrete integration. In *Proc. of the 29th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2013.
- [8] Stefano Ermon, Carla P. Gomes, Ashish Sabharwal, and Bart Selman. Taming the curse of dimensionality: Discrete integration by hashing and optimization. In *Proc. of the 30th International Conference on Machine Learning (ICML)*, 2013.
- [9] Stefano Ermon, Carla P. Gomes, Ashish Sabharwal, and Bart Selman. Low-density parity constraints for hashing-based discrete integration. In *Proc. of the 31st International Conference on Machine Learning (ICML)*, pages 271–279, 2014.
- [10] V. Gogate and R. Dechter. SampleSearch: Importance sampling in presence of determinism. *Artificial Intelligence*, 175(2):694–729, 2011.
- [11] Leslie Ann Goldberg and Mark Jerrum. The complexity of ferromagnetic ising with local fields. *Combinatorics, Probability and Computing*, 16(01):43–61, 2007.
- [12] Carla P. Gomes, A. Sabharwal, and B. Selman. Model counting: A new strategy for obtaining good bounds. In *Proc. of the 21st National Conference on Artificial Intelligence (AAAI)*, pages 54–61, 2006.
- [13] Carla P. Gomes, Willem Jan van Hoeve, Ashish Sabharwal, and Bart Selman. Counting CSP solutions using generalized XOR constraints. In *Proc. of the 22nd National Conference on Artificial Intelligence (AAAI)*, 2007.
- [14] Mark Jerrum and Alistair Sinclair. Polynomial-time approximation algorithms for the ising model. *SIAM Journal on computing*, 22(5):1087–1116, 1993.
- [15] Mark Jerrum and Alistair Sinclair. The markov chain monte carlo method: An approach to approximate counting and integration. In *Approximation Algorithms for NP-hard Problems*, pages 482–520. PWS Publishing, Boston, MA, 1997.
- [16] Michael I. Jordan, Z. Ghahramani, Tommi Jaakkola, and L.K. Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.
- [17] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT Press, 2009.
- [18] Alex Kulesza and Fernando Pereira. Structured learning with approximate inference. In *Advances in neural information processing systems*, pages 785–792, 2007.
- [19] N.N. Madras. *Lectures on Monte Carlo Methods*. American Mathematical Society, 2002.
- [20] Joris M. Mooij. libDAI: A free and open source C++ library for discrete approximate inference in graphical models. *Journal of Machine Learning Research*, 11:2169–2173, August 2010.
- [21] Radford M Neal. Annealed importance sampling. *Statistics and Computing*, 11:125–139, 2001.
- [22] Man-Suk Oh and James O. Berger. Adaptive importance sampling in monte carlo integration. *Journal of Statistical Computation and Simulation*, 41:143–168, 1992.
- [23] Dan Roth. On the hardness of approximate reasoning. *Artificial Intelligence*, 82(1):273–302, 1996.
- [24] Ruslan Salakhutdinov and Geoffrey E. Hinton. Deep Boltzmann machines. In *Proc. of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2009.
- [25] Ruslan Salakhutdinov and Iain Murray. On the quantitative analysis of deep belief networks. In *Proc. of the 25th International Conference on Machine Learning (ICML)*, 2008.
- [26] Sunita Sarawagi and Rahul Gupta. Accurate max-margin training for structured output spaces. In *Proceedings of the 25th international conference on Machine learning*, pages 888–895. ACM, 2008.
- [27] Dale Schuurmans. Greedy importance sampling. In *Advances in Neural Information Processing Systems (NIPS)*, 2000.
- [28] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Large margin methods for structured and interdependent output variables. In *Journal of Machine Learning Research*, pages 1453–1484, 2005.
- [29] Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.
- [30] Chun-Nam John Yu and Thorsten Joachims. Learning structural svms with latent variables. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1169–1176. ACM, 2009.

---

# Progressive Abstraction Refinement for Sparse Sampling

---

Jesse Hostetler and Alan Fern and Thomas Dietterich  
Department of Electrical Engineering and Computer Science  
Oregon State University  
{hostetje, afern, tgd}@eecs.oregonstate.edu

## Abstract

Monte Carlo tree search (MCTS) algorithms can encounter difficulties when solving Markov decision processes (MDPs) in which the outcomes of actions are highly stochastic. This stochastic branching can be reduced through state abstraction. In online planning with a time budget, there is a complex tradeoff between loss in performance due to overly coarse abstraction versus gain in performance from reducing the problem size. Coarse but unsound abstractions often outperform sound abstractions for practical budgets. Motivated by this, we propose a progressive abstraction refinement algorithm that refines an initially coarse abstraction during search in order to match the abstraction to the sample budget. Our experiments show that the algorithm combines the strong performance of coarse abstractions at small sample budgets with the ability to exploit larger budgets for further performance gains.

## 1 INTRODUCTION

When solving planning problems with a time budget, choosing the right representation is crucial. The native representation is often too detailed. It may treat many situations as distinct that are actually similar, causing the planner to spend its limited budget planning for irrelevant contingencies and fail to discover the long term consequences of actions. We should ignore details when deliberation time is limited, but given more time we should increase the level of detail to make better decisions. We consider how to design online planning algorithms that benefit from this type of progressive attention to detail.

Our approach is based on the Monte Carlo tree search (MCTS) paradigm. MCTS methods [Browne et al., 2012] select an action in a given state by constructing a lookahead search tree using a domain simulator. Because MCTS

methods plan for only one state at a time, their asymptotic sample complexity is generally independent of the size of the state space. Rather, they are limited by the search depth achievable with a given time budget, which must be deep enough to estimate the quality of actions in the current state.

A full expectimax tree of depth  $d$  has of the order  $O(|\mathcal{A}| \cdot B)^d$  nodes, where  $|\mathcal{A}|$  is the size of the action set and  $B$  is the number of possible outcomes per action. To achieve a larger value of  $d$ , it is necessary to limit branching in the tree. One way is to reduce  $|\mathcal{A}|$ , an approach taken in previous works such as [Pinto and Fern, 2014]. The other approach, and the one we consider, is to reduce  $B$ .

MCTS algorithms limit  $B$  by selective sampling. Sparse sampling (SS) algorithms [Kearns et al., 2002] limit  $B$  to a constant. Trajectory sampling (TS) algorithms like UCT [Kocsis and Szepesvári, 2006] focus exploration toward actions with higher estimated values, so that  $B$  and  $d$  are small in non-optimal subtrees. In this paper, we investigate a version of sparse sampling that employs state abstraction to further reduce  $B$ .

We limit ourselves to *state aggregation* abstractions, in which the ground states are partitioned into a set of aggregate states. Classes of aggregation abstractions that guarantee bounded performance loss in tree search algorithms have been constructed based on the value function [Hostetler et al., 2014] and on bisimilarity [Jiang et al., 2014]. But while performance loss bounds guarantee asymptotic accuracy, practical time budgets may preclude running a search to convergence. Thus the accuracy and size of the abstraction interact in a complex way to determine the actual search performance. Small budgets call for coarser abstractions, while large budgets can support finer abstractions [Jiang et al., 2014].

Our observation from experience with state abstraction in MCTS has been that search with the coarsest possible abstraction — the abstraction that maps all states to a single equivalence class — often substantially outperforms search in the ground state space for moderate search budgets. Yet

the optimal policy is seldom representable in this abstract space, and there comes a point at which search with such an abstraction cannot exploit further increases in the budget.

Our main contribution is an algorithm — PARSS — that exploits the strengths of coarse abstractions while retaining the convergence and optimality guarantees of search in the ground state space. We achieve this by beginning with a coarse abstraction and refining it during search to create a more detailed abstraction, thus allowing performance to continue to improve after the point where search with the starting abstraction would have reached a plateau. We prove that with suitable implementation choices, PARSS converges to the same result and with the same worst-case sample complexity as a sparse sampling search over the ground state space. Finally, we conduct experiments demonstrating the strength of very coarse abstractions in anytime online planning and the ability of PARSS to further improve upon their performance.

## 2 BACKGROUND

We consider a Monte Carlo tree search (MCTS) algorithm for online planning in Markov decision processes (MDPs). A discounted MDP is a 5-tuple  $M = \langle \mathcal{S}, \mathcal{A}, P, R, \gamma \rangle$ , where  $\mathcal{S}$  and  $\mathcal{A}$  are finite sets of states and actions,  $P(s'|s, a)$  is the transition function,  $R(s)$  gives the instantaneous reward in  $s$ , and  $\gamma \in [0, 1]$  is the discount factor. Note that undiscounted problems (ie.  $\gamma = 1$ ) pose no problem in our online planning setting.

Tree search algorithms construct sampled approximations of the *expectimax tree* over  $M$  rooted at the current state  $s_0$ . The expectimax tree itself defines an MDP over state-action histories as follows. Let  $\mathcal{H}(M, s_0) = \{s_0\} \times \mathcal{A} \times \mathcal{S} \times \dots$  be the set of state-action histories in  $M$  starting in  $s_0$ . We write  $\mathcal{H}^n$  for the set of histories of length  $n$ . Given a history  $h = s_0 a_1 s_1 \dots a_n s_n \in \mathcal{H}^n$ , we write  $s(h) \equiv s_n$  and  $a(h) \equiv a_n$  for the final state and final action in the history,  $p(h) \equiv s_0 a_1 s_1 \dots a_{n-1} s_{n-1}$  for the prefix of the history, and  $\ell(h) \equiv n$  for the length of the history. Using this notation, we overload the  $P$  and  $R$  functions to apply to histories by defining  $P(h'|h, a) = \mathbf{1}_{p(h')=h} \mathbf{1}_{a(h')=a} P(s'|s(h), a)$  and  $R(h) = R(s(h))$ . The *Tree MDP* induced by a general MDP  $M$  and rooted at  $h_0 = s_0$  is the tuple  $T(M, s_0) = \langle \mathcal{H}, \mathcal{A}, P, R, h_0, \gamma \rangle$ .

A solution of a Tree MDP is a policy  $\pi : \mathcal{H} \rightarrow \mathcal{A}$  mapping histories to actions. The *value* of a policy is given by the value function  $V^\pi(h) = R(h) + \sum_{h' \in \mathcal{H}} P(h'|h, \pi(h)) V^\pi(h')$ . A policy is *optimal* if  $V^\pi(h) = V^*(h)$ , where  $V^*(h)$  is the optimal value function,  $V^*(h) = R(h) + \max_{a \in \mathcal{A}} \sum_{h' \in \mathcal{H}} P(h'|h, a) V^*(h')$ . Equivalently, an optimal policy  $\pi^*$  is such that  $\pi^*(h) = \arg \max_{a \in \mathcal{A}} Q^*(h, a)$ , where  $Q^*(h, a) = R(h) + \sum_{h' \in \mathcal{H}} P(h'|h, a) \max_{a' \in \mathcal{A}} Q^*(h', a')$ .

## 2.1 SPARSE SAMPLING

Our proposed algorithm derives from the sparse sampling (SS) algorithm [Kearns et al., 2002]. SS estimates  $Q^*(h_0, a)$  by constructing a sampled approximation of the Tree MDP rooted at  $h_0$ . An SS tree of depth  $d$  and width  $C$  defines a finite horizon estimate  $Q^d(h, a) = R(h) + \frac{\gamma}{C} \sum_{h' \in k(h, a)} \max_{a' \in \mathcal{A}} Q^{d-1}(h', a')$  of  $Q^*$ , with terminal values  $Q^0(h, a) = R(h)$ . Each  $k(h, a)$  is a collection of  $C$  iid samples  $h' \sim P(\cdot|h, a)$ , possibly containing duplicates. The greedy policy with respect to  $Q^d$ ,  $\pi_{SS}(h) = \arg \max_{a \in \mathcal{A}} Q^d(h, a)$ , achieves bounded suboptimality with sample complexity independent of the size of the state space. Our algorithm (Section 4) is based on Forward Search Sparse Sampling (FSSS) [Walsh et al., 2010], an SS algorithm that incorporates pruning.

## 2.2 STATE ABSTRACTION

We consider the simplest form of state abstraction — state aggregation — in which the abstract states are the members of a partition of the ground state space. Given a Tree MDP  $T = \langle \mathcal{H}, \mathcal{A}, P, R, h_0, \gamma \rangle$ , a state abstraction of  $T$  is an equivalence relation  $\chi$  on the set  $\mathcal{H}$ . The abstraction relation induces an abstract state space  $\mathcal{H}/\chi$  whose members are the equivalence classes of  $\mathcal{H}$  with respect to  $\chi$ . We will write  $T/\chi$  as shorthand for  $\langle \mathcal{H}/\chi, \mathcal{A}, P, R, h_0, \gamma \rangle$ . The equivalence class of  $h$  with respect to  $\chi$  is denoted  $[h]_\chi$ , and we say that two states  $h$  and  $g$  are equivalent with respect to  $\chi$ , denoted  $h \simeq_\chi g$ , if  $[h]_\chi = [g]_\chi$ . The abstraction relation for a Tree MDP must be such that  $h \simeq_\chi g \Rightarrow p(h) \simeq_\chi p(g)$  to ensure that  $T/\chi$  is also a Tree MDP.

An abstraction  $\chi$  is *sound* if there is an optimal policy  $\pi^*$  over  $T$  such that  $h \simeq_\chi g \Rightarrow \pi^*(h) = \pi^*(g)$  for all  $h, g \in \mathcal{H}$ . Classes of aggregation abstractions that are sound or that guarantee bounded performance loss in MCTS algorithms have been studied by Hostetler et al. [2014] and Jiang et al. [2014].

## 2.3 ABSTRACTION REFINEMENT

State equivalence abstractions form a complete lattice ordered by abstraction granularity.

**Definition 1.** Abstraction  $\psi$  is *finer* than  $\chi$ , denoted  $\psi \preceq \chi$ , if  $h \simeq_\psi g \Rightarrow h \simeq_\chi g$ . If in addition  $\psi \neq \chi$ , then  $\psi$  is *strictly finer* than  $\chi$ , denoted  $\psi \prec \chi$ .

The finest abstraction is the *bottom* or *ground* abstraction  $\perp$ , which maps all states to singleton sets,  $[h]_\perp = \{h\} \forall h$ . The coarsest abstraction is the *top* abstraction  $\top$ , which maps all ground states of the same length to the same abstract state,  $[h]_\top = \mathcal{H}^{\ell(h)} \forall h$ . Searching in the abstract problem  $T/\top$  amounts to searching for the best open-loop policy in  $T$ , while searching in  $T/\perp$  is equivalent to searching in the ground space.



The lattice structure of equivalence abstractions ensures that by iteratively constructing strict refinements of any initial abstraction we will eventually reach the bottom abstraction  $\perp$ , which is trivially sound. Furthermore, this property does not depend on how the refinements are chosen.

### 3 MOTIVATING EXAMPLE: THE SAVING PROBLEM

To illustrate the type of problem structure that motivates our approach to state abstraction, we created a toy problem called the Saving problem (Figure 1). The Saving problem is an episodic task in which the agent must accumulate wealth by choosing to either *save*, *invest*, or *borrow* at each time step. The *save* action always yields an immediate reward of 1. The *borrow* action takes out a “loan”, which gives an immediate reward of 2 and starts a countdown timer  $t_b$  from  $T_b$  to 0. The agent cannot *borrow* again while  $t_b > 0$ . When  $t_b$  reaches 0, the agent receives a reward of  $-3$ , representing repaying the loan with interest. Thus the true value of *borrow* is  $-1$ , unless the episode will end before the loan is repaid. The *invest* action gives 0 immediate reward, but gives the agent the right to take the *sell* action sometime during the next  $T_i$  time steps. The *sell* action gives a reward  $price(t)$  if executed at time  $t$ , where each  $price(t) \sim \text{DiscreteUniform}\{p_{\min}, p_{\max}\}$  for  $p_{\min}, p_{\max} \in \mathbb{Z}$ . The agent can have only one investment at a time.

We instantiate the Saving problem with  $p_{\min} = -4$ ,  $p_{\max} = 4$ ,  $T_i = 4$ , and  $T_b = 4$ . With these parameters, *invest* is nearly always optimal, but only if the agent takes advantage of the investment period  $T_i$  in order to sell the investment for more than  $\mathbb{E}[price] = 0$ . *Borrow* is almost always the worst action, but the agent must search to a depth of at least  $T_b$  to discover its negative consequences.

Let us consider how the two extremes of state abstraction interact with the Saving problem, beginning with the ground abstraction  $\perp$ . To avoid falling into the *borrow* trap, the search must reach a depth of at least  $T_b$ . A full expectimax tree of depth  $T_b$  has size of the order  $O((|\mathcal{A}|B)^{T_b})$ , where  $B = p_{\max} - p_{\min} + 1$  is the stochastic branching due to the random fluctuation of *price*. If we are restricted to a sample budget  $k \ll (|\mathcal{A}|B)^{T_b}$ , then the constructed tree will be very incomplete, and the agent may accidentally choose to *borrow* or not to *invest* due to sampling variance.

At the other extreme, when searching with the top abstraction  $\top$ , the size of the abstract tree is of the order  $O(|\mathcal{A}|^{T_b})$ . This tree is much smaller in practical terms, so searches will tend to give lower-variance estimates and *borrow* should be chosen less often. On the other hand, since search with  $\top$  cannot discriminate between states, the value of *invest* will be estimated as  $\mathbb{E}[price] = 0$ . This estimate is less than 2, which is the opportunity cost of doing *invest* and *sell* instead of doing *save* twice. Thus search with  $\top$

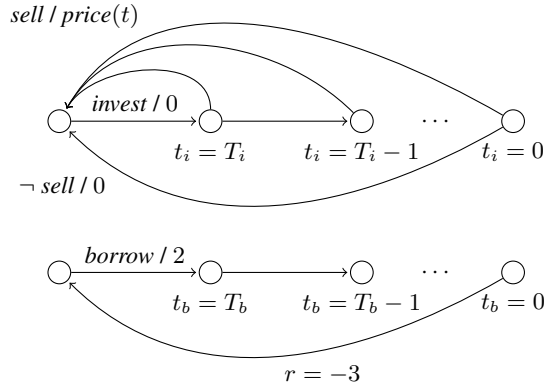


Figure 1: Schematic diagram of the *invest* and *borrow* actions in the Saving problem. Edges labeled with an action show its immediate reward.

will incorrectly choose to *save* rather than *invest*. This failure mode of open loop replanning was noted by Weinstein and Littman [2012].

These two extremes illustrate an important point: the appropriate abstraction depends on the sample budget. The top abstraction  $\top$  is superior for small budgets. Although  $\top$  is unsound in this domain, the search is operating in a much smaller state space, resulting in lower variance and less chance of incorrectly choosing *borrow*. Conversely,  $\perp$  is best with a large sample budget, since  $\perp$  is sound whereas coarser abstractions might not be sound. With intermediate sample budgets, it is not clear how to determine the appropriate abstraction granularity.

### 4 PROGRESSIVE ABSTRACTION REFINEMENT

The difficulties illustrated in the Saving problem motivate our proposed algorithm. We construct an abstract sparse sampling algorithm that begins with the coarsest abstraction  $\top$  and progressively refines the abstraction during search. If the algorithm is interrupted early in the search, it gives an answer based on a coarse but inaccurate abstraction. As more samples accumulate, the abstraction is refined and the abstract search transforms smoothly into a search over the ground state space.

The algorithm is built on top of the Forward Search Sparse Sampling (FSSS) algorithm of Walsh et al. [2010]. We first describe how to modify FSSS to construct a search tree over the abstract state space induced by a fixed abstraction. We then “wrap” the abstract search in a progressive abstraction refinement procedure.

In our algorithm descriptions, we treat each history  $h$  as an object, in the sense that two histories  $h$  and  $g$  are dis-

tinct even if they describe the same sequence of states and actions. Abstract states  $H$  are collections of ground histories  $h$ , and may contain duplicates. The immediate reward for an abstract state is denoted  $\mathcal{R}(H)$  and is equal to the average reward over its constituent ground states,  $\mathcal{R}(H) = \frac{1}{|H|} \sum_{h \in H} R(h)$ . We also assume the availability of admissible value bounds  $V_{\min}$  and  $V_{\max}$  such that  $V_{\min} \leq V^\pi(h) \leq V_{\max}$  for all ground states  $h \in \mathcal{H}$  and for all policies  $\pi$  over  $\mathcal{H}$ .

#### 4.1 ABSTRACTION IN SPARSE SAMPLING

Forward Search Sparse Sampling (FSSS) [Walsh et al., 2010] is an enhancement of ordinary sparse sampling (SS) that incorporates pruning based on upper and lower bounds on the values of subtrees. It provides the same performance guarantees as SS and often does less computation.

Abstract FSSS (AFSSS; Algorithm 1) is a straightforward extension of FSSS. AFSSS constructs an FSSS tree over abstract states. The abstract tree encapsulates a tree of ground states, whose structure is defined by collections of ground successors  $k(h, a)$ . Each abstract *state node* is a collection  $H = \{h_i\}$  of ground states. Associated with each abstract state node are an upper and a lower value bound  $U(H)$  and  $L(H)$ <sup>1</sup> and a visit count  $n(H)$ . Each state node  $H$  such that  $n(H) > 0$  has an *action node* successor  $Ha$  for each  $a \in \mathcal{A}$ . Action nodes have associated value bounds  $U(H, a)$  and  $L(H, a)$ , abstraction relations  $\chi(H, a)$ , and abstract successor sets  $K(H, a)$ . The visit count  $n(H, a)$  for an action node  $Ha$  is equal to the number of ground successors of  $Ha$ ,  $n(H, a) = \sum_{h \in H} |k(h, a)|$ .

The inputs to AFSSS are an abstract FSSS tree  $\mathcal{T}$ , sampling width  $C$ , and maximum depth  $d$ . Like FSSS, AFSSS proceeds in a series of top-down trials that each traverse a path from the root node to a leaf state node. When extending a path, the algorithm chooses action nodes optimistically (Line 11), and chooses state nodes with the largest gap between  $U$  and  $L$  (Line 12). If the path reaches an unvisited state node (Line 9), that node is *expanded* by initializing and sampling its action node successors. The backup operation (Line 28) combines the average immediate reward over ground states with the discounted future return bounds over abstract states weighted by their empirical frequency.

When sampling an action node  $Ha$  (Line 22), the algorithm must ensure that  $n(H, a) \geq C$  to satisfy the sparse sampling property. We accomplish this by sampling ground successors  $h' \sim P(\cdot|h, a)$  for each ground state  $h \in H$  and adding them to the ground successor collections  $k(h, a)$  until  $|k(h, a)| = \lceil C/|H| \rceil$  for all  $h \in H$ . Note that this sampling method will sometimes draw more than  $C$  samples

<sup>1</sup>As in FSSS, these quantities bound the value estimate of the full SS tree conditioned on the samples so far. The value of an abstract state is a particular weighted average of ground values. See [Hostetler et al., 2014] for details.

---

#### Algorithm 1 Abstract Forward Search Sparse Sampling

---

```

1: procedure AFSSS( $\mathcal{T} = \langle K, L, U, H_0, \chi \rangle, C, d, \chi_0$ )
2:   global  $K, L, U, H_0, \chi, C, \chi_0$ 
3:   while time remains and not converged do
4:     VISIT( $H_0, d$ )
5:   procedure VISIT( $H, d$ )
6:     if  $H$  is terminal or  $d = 0$  then
7:        $L(H) \leftarrow \mathcal{R}(H), U(H) \leftarrow \mathcal{R}(H)$ 
8:     else
9:       if  $n(H) = 0$  then EXPAND( $H, \chi_0$ )
10:       $n(H) \leftarrow n(H) + 1$ 
11:       $a^* \leftarrow \arg \max_a U(H, a)$ 
12:       $H^* \leftarrow \arg \max_{H' \in K(H, a^*)} [U(H') - L(H')]$ 
13:      VISIT( $H^*, d - 1$ )
14:      BACKUP( $H, a^*$ )
15:      BACKUP( $H$ )
16:   procedure EXPAND( $H$ )
17:     for all  $a \in \mathcal{A}$  do
18:        $\chi(H, a) \leftarrow \chi_0(H, a)$ 
19:        $(L(H, a), U(H, a)) \leftarrow (V_{\min}, V_{\max})$ 
20:       SAMPLE( $H, a$ )
21:        $(L(H'), U(H')) \leftarrow (V_{\min}, V_{\max}) \forall H' \in K(H, a)$ 
22:   procedure SAMPLE( $H, a$ )
23:     for all  $h \in H$  do
24:       while  $|k(h, a)| < \lceil C/|H| \rceil$  do
25:          $h' \sim P(\cdot|h, a)$ 
26:          $k(h, a) \leftarrow k(h, a) \cup \{h'\}$ 
27:        $K(H, a) \leftarrow [\bigcup_{h \in H} k(h, a)] / \chi(H, a)$ 
28:   procedure BACKUP( $H, a$ )
29:      $L(H, a) \leftarrow \mathcal{R}(H) + \gamma \sum_{H' \in K(H, a)} \frac{|H'|}{n(H, a)} L(H')$ 
30:      $U(H, a) \leftarrow \mathcal{R}(H) + \gamma \sum_{H' \in K(H, a)} \frac{|H'|}{n(H, a)} U(H')$ 
31:   procedure BACKUP( $H$ )
32:      $L(H) \leftarrow \max_a L(H, a)$ 
33:      $U(H) \leftarrow \max_a U(H, a)$ 

```

---

for an abstract action node  $Ha$ . It is crucial that sampling is done in this way to allow the abstraction refinement algorithm we will build on top of AFSSS (Section 4.2) to have the same performance guarantees as FSSS.

AFSSS terminates when the time budget is exceeded or the tree has converged (Line 3). The tree has converged if

$$L(H_0, a^*) \geq \max_{a \neq a^*} U(H_0, a) \quad (1)$$

where  $a^* = \arg \max_{a \in \mathcal{A}} L(H_0, a)$ .

#### 4.2 PROGRESSIVE ABSTRACTION REFINEMENT FOR SPARSE SAMPLING

Our proposed algorithm (Algorithm 3) begins by building an abstract FSSS tree with respect to  $\mathcal{T}$ . After building the abstract tree, it begins to refine the abstraction, and contin-

---

**Algorithm 2** A generic abstraction refinement procedure

---

```
1: procedure PAR( $\mathcal{T} = \langle K, L, U, H_0, \chi \rangle$ )
2:   Let  $Ha = \text{SELECT}(\mathcal{T})$ 
3:   if  $Ha \neq \emptyset$  then
4:      $\chi(H, a) \leftarrow \text{REFINE}(\chi(H, a))$ 
5:     SPLIT( $H, a, \chi$ )
6:     UPDATETREE( $H, a$ )
```

---

ues until there are no more useful refinements to perform. We call the algorithm Progressive Abstraction Refinement for Sparse Sampling (PARSS).

PARSS combines AFSSS with an instantiation of the generic refinement procedure PAR described in Algorithm 2. The PAR procedure consists of four steps. The SELECT function either returns an action node  $Ha$  whose associated abstraction relation  $\chi(H, a)$  should be refined, or indicates that no refinement is to be done. The REFINE procedure is then called on the selected abstraction relation. After refinement, the tree is SPLIT recursively to respect the new abstraction. Finally, UPDATETREE re-computes the tree statistics as necessary. The implementations of SPLIT (Line 10) and UPDATETREE (Line 20) for AFSSS are straightforward. The remaining operations, SELECT and REFINE, are described in the next two sections.

After each PAR operation, PARSS calls AFSSS on the refined tree. This is necessary because refinement may have changed the value bounds of the root node such that the tree no longer satisfies the convergence criterion.

#### 4.2.1 Implementing SELECT

To ensure soundness, the SELECT operation must eventually select all action nodes  $Ha$  such that refining  $Ha$  could change the root action choice. A selection mechanism that guarantees this is said to be *complete*.

**Definition 2.** A SELECT mechanism is *complete* if it returns an action node  $Ha$ , whenever such an  $Ha$  exists, such that:

1.  $\chi(H, a) \succ \perp$ .
2.  $U(H, a) < V_{\max}$

These conditions ensure that refining  $\chi(H, a)$  could potentially alter the choice of root action. Condition (2.2) excludes action nodes in which  $U(H, a) = V_{\max}$ , since in this case refining below  $Ha$  cannot increase  $U(H, a)$  and thus cannot affect the root action choice. This situation arises when nodes have been generated by the EXPAND operation in AFSSS, but have not been visited yet.

The requirements of Definition 2 place few constraints on the order in which nodes are selected. There are several heuristic reasons to prefer refining near the root first. The

---

**Algorithm 3** Progressive Abstraction Refinement for SS

---

```
1: procedure PARSS( $h_0, C, d$ )
2:   Let  $\mathcal{T} = \langle K, L, U, H_0, \chi \rangle$  where
3:      $K(H_0, a) = \emptyset \quad \forall a \in \mathcal{A}$ ,
4:      $L(H_0) = V_{\min}, U(H_0) = V_{\max}$ ,
5:      $H_0 = \{h_0\}, \chi = \top$ .
6:   AFSSS( $\mathcal{T}, C, d, \top$ )
7:   while time remains and some  $\chi(H, a) \succ \perp$  do
8:     PAR( $\mathcal{T}$ )
9:     AFSSS( $\mathcal{T}, C, d, \top$ )
10:  procedure SPLIT( $H, a, \chi$ )
11:  if  $H$  is a leaf then return
12:  Let  $K' = \emptyset$  ▷ New abstract successor set
13:  for all  $H' \in K(H, a)$  do
14:    Let  $\mathcal{G}' = H' / \chi(H, a)$  ▷ Refined partition
15:    for all  $\langle G', a' \rangle \in \mathcal{G}' \times \mathcal{A}$  do
16:       $K' \leftarrow K' \cup \{G'\}$ 
17:       $\chi(G', a') \leftarrow \chi(H', a')$  ▷ Copy old relation
18:      SPLIT( $G', a', \chi$ )
19:   $K(H, a) \leftarrow K'$  ▷ Overwrite old successor set
20:  procedure UPDATETREE( $H, a$ )
21:  for all  $H' \in K(H, a)$  do
22:    UPSAMPLE( $H'$ )
23:  for  $t$  from 0 to  $\ell(H)$  do ▷ Backup path to root
24:    for all  $a \in \mathcal{A}$  do BACKUP( $H, a$ )
25:    BACKUP( $H$ )
26:    Let  $H = p(H)$ 
27:  procedure UPSAMPLE( $H$ )
28:  if  $H$  is a leaf then
29:     $L(H) \leftarrow \mathcal{R}(H), U(H) \leftarrow \mathcal{R}(H)$ 
30:  else if  $n(H) > 0$  then
31:    for all  $a \in \mathcal{A}$  do
32:      SAMPLE( $H, a$ )
33:    for all  $H' \in K(H, a)$  do UPSAMPLE( $H'$ )
34:    BACKUP( $H, a$ )
35:    BACKUP( $H$ )
```

---

most important is that actions near the root are part of more different policies than actions near the leaves. Since the value of an action node only affects the root value if that action is part of the optimal policy, refining nodes that are members of more policies makes it more likely that the refinement will affect the root value. In discounted problems, nodes at shallow depths are also less affected by discounting. These observations suggest that a breadth-first ordering is a reasonable choice.

In our experiments, we used a randomized breadth-first strategy to choose the next node to refine. Our SELECT implementation is divided into subtree selection and node selection phases. First, a subtree  $H_0a$  that is not fully refined is chosen uniformly at random. Then we find the shallowest depth  $d$  at which some descendent of  $H_0a$  satisfies the

conditions of Definition 2, and return one such descendent at depth  $d$  uniformly at random.

#### 4.2.2 Implementing REFINE

Due to the lattice structure of partition abstractions, repeated refinements will eventually yield the ground abstraction  $\perp$ , provided the refinements are strict. Thus, we require that REFINE produces *strict* refinements, to guarantee that the refinement process continues to make progress.

**Definition 3.** REFINE is a *strict refinement function* if  $\text{REFINE}(\chi) \prec \chi$ .

The best choice of REFINE implementation will depend on the problem being solved. In our experiments, we tried the following two variations.

**Random Refinement.** The simpler approach, `REFINERANDOM`, first chooses the largest set  $H'$  in the partition induced by  $\chi(H, a)$ . It then randomly permutes the equivalence classes in  $H'/\perp$  and greedily divides them into two sets of approximately equal size. This option is fast to compute but does not exploit structure in the ground state space.

**Tree-based Refinement.** If we have access to a set of features  $\{\phi_i(h)\}$  for each state, we can take a more sophisticated approach. `REFINEDT` is based on an incrementally-constructed decision tree. Each abstraction relation  $\chi(H, a)$  is defined by a decision tree  $D$ . The leaves of  $D$  define the members of a partition of the successors of  $Ha$ . Interior nodes are labeled with a feature  $i$  and a threshold  $\theta$ . The refinement operation chooses the largest leaf node  $N$  of  $D$  and adds a new split to  $D$  dividing  $N$  into two new sets  $X$  and  $Y$ , choosing splits greedily to maximize an evaluation function  $f(X, Y)$ .

The evaluation function  $f$  can be designed to encourage desired properties in the partitions. For example, if  $\chi$  is such that for all  $H \in \mathcal{H}/\chi$ , all members of  $H$  have the same optimal action and the same optimal value, then  $\chi$  is sound in sparse sampling [Hostetler et al., 2014]. This condition is called  $a^*$ -irrelevance [Li et al., 2006]. We define an evaluation function that encourages  $a^*$ -irrelevance using upper bounds  $u(h)$  and  $u(h, a)$  for ground state values. These can be computed along with the bounds for the abstract states during the `BACKUP` step (Algorithm 1, Line 28).

Using these bounds on the ground states, we define the evaluation function

$$f(X, Y) = |\bar{u}(X) - \bar{u}(Y, a^*)| + |\bar{u}(Y) - \bar{u}(X, b^*)|,$$

where  $\bar{u}(H) = \frac{1}{|H|} \sum_{h \in H} u(h)$ ,  $\bar{u}(H, a) = \frac{1}{|H|} \sum_{h \in H} u(h, a)$ ,  $a^* = \arg \max_{a \in \mathcal{A}} \bar{u}(X, a)$  and  $b^* = \arg \max_{b \in \mathcal{A}} \bar{u}(Y, b)$ . Splits that maximize  $f$  will tend to put ground states that have different optimal actions or different optimal values into different abstract states.

### 4.3 ANALYSIS OF PARSS

The PARSS algorithm can be viewed as a different way of orchestrating the sampling of a sparse tree. If run to termination, it provides the same performance guarantees with the same sample complexity as ordinary sparse sampling.

**Definition 4.** An abstract search tree  $\mathcal{T} = \langle K, L, U, H_0 \rangle$  is an *abstract FSSS tree with respect to  $\chi$* , or an *AFSSS( $\chi$ ) tree* in shorthand, if

1. For each abstract state node  $H, \forall h, g \in H, h \simeq_\chi g$ ;
2. For every abstract state node  $H$  such that  $n(H) > 0$ ,  $n(H, a) \geq C$  for all  $a \in \mathcal{A}$ ,
3. All value bounds  $L$  and  $U$  are admissible (Section 4.1),
4.  $\mathcal{T}$  satisfies the AFSSS convergence criterion (1).

One can easily verify that the output of AFSSS is an abstract FSSS tree.

**Proposition 1.** Consider a PARSS implementation where the `SELECT` and `REFINE` operations satisfy the conditions of Definitions 2 and 3. If the current search tree  $\mathcal{T}$  is an abstract FSSS tree with respect to abstraction  $\chi$ , then after one iteration of the loop in Algorithm 3, Line 7, the resulting tree  $\mathcal{T}'$  is an abstract FSSS tree with respect to an abstraction  $\psi$  such that  $\psi \prec \chi$ .

*Proof.* By assumption,  $\text{REFINE}(\chi(H, a))$  returns a new abstraction  $\psi$  such that  $\psi(H, a) \prec \chi(H, a)$ , and therefore  $\psi \prec \chi$ . The `SPLIT` operation partitions the subtree  $Ha$  according to  $\psi$ , establishing condition (4.1). The `UPSAMPLE` loop in `UPDATETREE` (Line 21) adds samples and performs backups in the subtree  $Ha$  to establish (4.2) and (4.3) for the subtree. Then values are backed up from  $Ha$  to the root node (Line 23), which establishes (4.3) for the rest of the tree. Finally, the call to `AFSSS` (Line 9) establishes convergence (4.4).  $\square$

**Proposition 2.** If PARSS does not exhaust its time budget, it terminates after drawing at most  $(|\mathcal{A}| \cdot C)^d$  samples from the transition function  $P$ , and the resulting search tree is an abstract FSSS tree with respect to  $\perp$ .

*Proof.* By Proposition 1, each iteration of the loop in Algorithm 3, Line 7 produces a strictly refined AFSSS tree. Due to the lattice structure of aggregation abstractions (Section 2.3), the abstraction relations  $\chi(H, a)$  will be equal to  $\perp$  for all  $H, a$  after a finite number of iterations. The tree at this point is an abstract FSSS tree with respect to  $\perp$ . The worst-case sample complexity occurs if all abstract nodes  $H$  in the fully-refined tree are singletons.  $\square$

**Proposition 3.** PARSS achieves the same error bounds and sample complexity as ordinary sparse sampling.

*Proof.* Proposition 2 establishes that PARSS yields an AFSSS( $\perp$ ) tree  $\mathcal{T}$  with the same worst-case sample complexity as SS (ie.  $O((|\mathcal{A}| \cdot C)^d)$ ).  $\mathcal{T}$  is different from a ground FSSS tree in that states that are equal in the ground representation are aggregated in  $\mathcal{T}$ . Because the FSSS pruning mechanism is sound [Walsh et al., 2010],  $\mathcal{T}$  achieves the same error bounds as an SS tree in which identical states are aggregated. The error bounds for sparse sampling remain valid in this case [Kearns et al., 2002], thus the conclusion follows.  $\square$

## 5 RELATED WORK

The PARSS algorithm begins with an abstract tree search under abstraction  $\top$ , which as we have noted is equivalent to searching for an open loop policy. Several works have explored the use of open loop policies for value estimation. Weinstein and Littman [2012] applied this idea in continuous action MDPs, drawing on theory developed by Bubeck and Munos [2010]. Weinstein and Littman [2013] later developed a related algorithm with a different optimization mechanism and applied it to legged locomotion tasks. Hauser [2011] used forward search with open loop policies to plan in partially observable continuous spaces.

Incremental construction of state space partitions starting from the top abstraction  $\top$  has been a common approach to abstraction discovery in MDPs, in algorithms such as the G algorithm [Chapman and Kaelbling, 1991], the PARTI-GAME algorithm [Moore and Atkeson, 1995], and the UTREE algorithm [McCallum, 1996]. The REFINEDT operation (Section 4.2.2) is similar to UTREE.

One closely related work is the Tree Learning Search (TLS) algorithm of Van den Broeck and Driessens [2011]. TLS contains all of the main ideas we use in PARSS, but applied in the UCT algorithm and targeted at continuous action spaces. Each state node of a TLS tree has an associated decision tree that represents a discretization of the action space. The decision trees are grown incrementally when data indicate that an existing action equivalence class leads to states with large variation in their values.

Another closely related work is that of Jiang et al. [2014], which describes a different abstraction refinement procedure for UCT. Their search proceeds in “batches” of samples. In between each batch, all of the abstraction relations are recomputed to satisfy a local approximate homomorphism criterion with respect to the sampled tree, and then the next batch of samples is drawn via search in the new abstract state space. A key difference from our approach is that their algorithm computes a particular abstraction with specified approximation bounds. With more samples, the computed abstraction becomes a better estimate of the target abstraction. In contrast, our method computes progressively finer abstractions as the sample budget increases.

Table 1: Best parameters for each algorithm. Parameter quality is measured by AUAC( $w_{\text{mag}}$ ).

| Algorithm  | Domain |     |     | Saving |     |     | Racetrack (S) |     |     | Racetrack (L) |     |     |
|------------|--------|-----|-----|--------|-----|-----|---------------|-----|-----|---------------|-----|-----|
|            | $B$    | $C$ | $d$ | $B$    | $C$ | $d$ | $B$           | $C$ | $d$ | $B$           | $C$ | $d$ |
| PARSS+DT   | -      | 5   | 5   | -      | 10  | 5   | -             | 20  | 4   | -             | 20  | 4   |
| PARSS+RAND | -      | 5   | 5   | -      | 10  | 5   | -             | 20  | 4   | -             | 20  | 4   |
| TOP        | -      | 5   | 5   | -      | 20  | 5   | -             | 20  | 4   | -             | 20  | 4   |
| GROUND     | -      | 2   | 5   | -      | 10  | 4   | -             | 10  | 4   | -             | 10  | 4   |
| RANDOM     | 2      | 5   | 5   | 2      | 20  | 4   | 2             | 20  | 4   | 2             | 10  | 4   |

| Algorithm  | Domain |     |     | Blackjack |     |     | Advising 1 |     |     | Advising 2 |     |     |
|------------|--------|-----|-----|-----------|-----|-----|------------|-----|-----|------------|-----|-----|
|            | $B$    | $C$ | $d$ | $B$       | $C$ | $d$ | $B$        | $C$ | $d$ | $B$        | $C$ | $d$ |
| PARSS+DT   | -      | 50  | 2   | -         | 5   | 2   | -          | 5   | 2   | -          | 5   | 2   |
| PARSS+RAND | -      | 50  | 2   | -         | 5   | 2   | -          | 5   | 2   | -          | 5   | 2   |
| TOP        | -      | 50  | 2   | -         | 5   | 2   | -          | 5   | 2   | -          | 5   | 2   |
| GROUND     | -      | 50  | 2   | -         | 2   | 2   | -          | 2   | 2   | -          | 2   | 2   |
| RANDOM     | 2      | 50  | 2   | 2         | 5   | 2   | 2          | 5   | 2   | 2          | 5   | 2   |

## 6 EXPERIMENTS

We evaluated PARSS with both REFINEDT and REFINERANDOM refinement procedures (“PARSS+DT” and “PARSS+RAND”; Section 4.2.2) in comparison to flat FSSS (“GROUND”), AFSSS with the top abstraction (“TOP”), and AFSSS with random abstractions of different fixed branching factors (“RANDOM”). We compared the anytime performance of the algorithms with both sample budgets and time budgets.

### 6.1 DOMAINS

Our test domains included the Saving problem described earlier (Section 3) as well as several other benchmark domains. We chose domains that exhibit varying amounts of stochastic branching and on which we suspected that the top abstraction would not be optimal.

**Racetrack.** Racetrack is the classic RL domain of Barto et al. [1995]. The agent controls a car in a grid world. The state specifies the car’s position and velocity, and the actions apply an acceleration  $a \in \{-1, 0, 1\} \times \{-1, 0, 1\}$  to the car. The objective is to reach a goal state in as few steps as possible without crashing into a wall. We incorporate stochasticity by making both components of the acceleration action subject independently to a random “slip” with probability 0.2, which causes 0 acceleration to be applied in that direction rather than the intended amount. We used both the “small” and “large” circuits of Barto et al. [1995].

**Spanish Blackjack.** Spanish Blackjack is a more complicated variation of the casino game Blackjack (or “21”). In Spanish Blackjack, the player may split to a total of up to 4 hands, may double down after splitting, may re-double the same hand up to a total of three times, and may hit after doubling. There are also bonuses for making 21 in 5 cards, 6 cards, or 7 or more cards, or with either 7,7,7 or 6,7,8.

**Academic Advising.** The Academic Advising domain [Guerin et al., 2012] was featured at the International Planning Competition at ICAPS in 2014. The agent must take

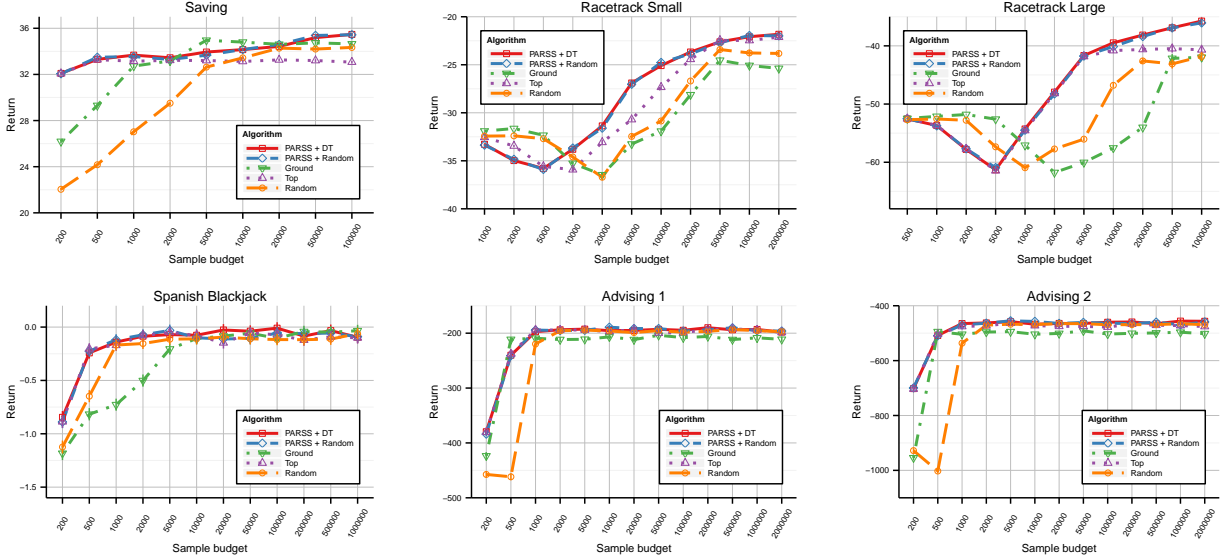


Figure 2: Performance vs. sample budget. Results are for the parameters that maximized  $AUAC(w_{\text{mag}})$ . Confidence intervals are shown, but are mostly smaller than the marker shapes except for Spanish Blackjack.

and pass all of the required courses in an academic program. The courses are linked by prerequisite relationships, and the chance of passing a course depends on how many of its prerequisites have been passed. We used MDP instances 1 and 2 from the IPC 2014.

In the IPC version of Advising, the agent can either *pass* or *fail* a course. We implemented a generalized problem that has integer grades in the range  $\{0, \dots, g\}$  to increase stochastic branching. We set  $g = 4$  and set 2 as the minimum passing grade for required courses.

## 6.2 METHODOLOGY

We evaluated each algorithm for several combinations of the  $C$  and  $d$  parameters. The range of  $d$  spanned 3 - 4 consecutive integers for each problem and  $C$  spanned 3 - 4 consecutive values in the sequence  $\{5, 10, 20, 50, 100, 200\}$ . Specific ranges were chosen based on pilot experiments.

For the sample budget experiments, we restricted the algorithms to a maximum number of samples from the transition function. We evaluated sample budgets in the sequence  $\{200, 500, 1000, 2000, \dots\}$  for each parameter combination. For each budget, we measured average return over 2000 to 10000 episodes, depending on the domain.

We report results for the combinations of parameters that maximized the weighted area under the anytime curve. Given a budget sequence  $\mathcal{B} = \{b_1, \dots, b_n\}$ , we calculate AUAC as

$$AUAC(w)(\mathcal{B}) = \sum_{i=2}^n w(b_i, b_{i-1}) \frac{\rho(b_i) + \rho(b_{i-1})}{2}, \quad (2)$$

where  $\rho(b)$  is the sample average of the return of the algorithm with budget  $b$  and  $w : \mathcal{B} \times \mathcal{B} \mapsto \mathbb{R}^{\geq 0}$  is a weight function. In our main experiment, we use the weight function  $w_{\text{mag}}(b_i, b_{i-1}) = \log b_i - \log b_{i-1}$ , which reflects a preference for good performance across orders of magnitude of budget.

Performance comparisons based on sample budgets can be misleading, since more-sophisticated algorithms do more work per sample. For those domains in which PARSS showed superior performance, we ran experiments with a time budget using the same parameters that achieved the best sample budget performance. The time budgets were measured in milliseconds and were drawn from the sequence  $\{10, 16, 25, 40, 63, 100, \dots\}$ .

## 6.3 RESULTS

The two variations of PARSS had superior anytime performance with a sample budget in Saving and in both Racetracks (Figure 2), and were equal to TOP on the other domains. TOP outperformed GROUND in all domains except Saving but it plateaued at a suboptimal value in Saving and Racetrack Large. The pattern of performance on Saving was as expected, with TOP plateauing while both variants of PARSS continued to improve and equaled the best performance of GROUND.

In Blackjack and both Advising domains, the  $AUAC(w_{\text{mag}})$  performance measure emphasized quick convergence, and the best  $C$  and  $d$  parameters were at the small end of their ranges (Table 6.1). TOP (and thus PARSS) converged more quickly than GROUND, and in both Advising domains also converged to a better value. PARSS did not diverge

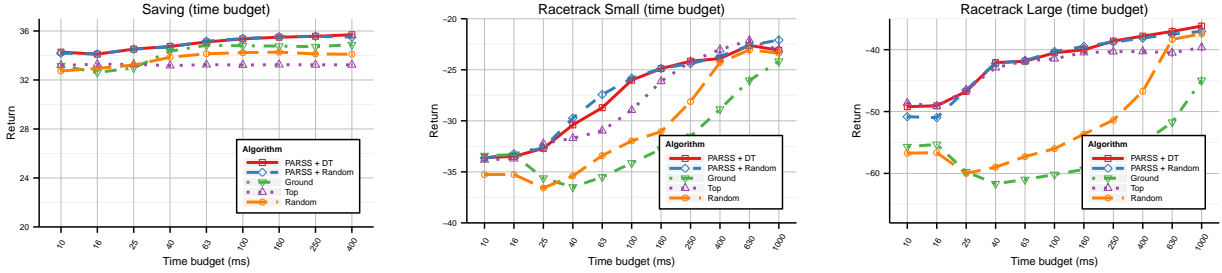


Figure 3: Performance vs. time budget. Parameter settings are the same as for the sample budget experiments.

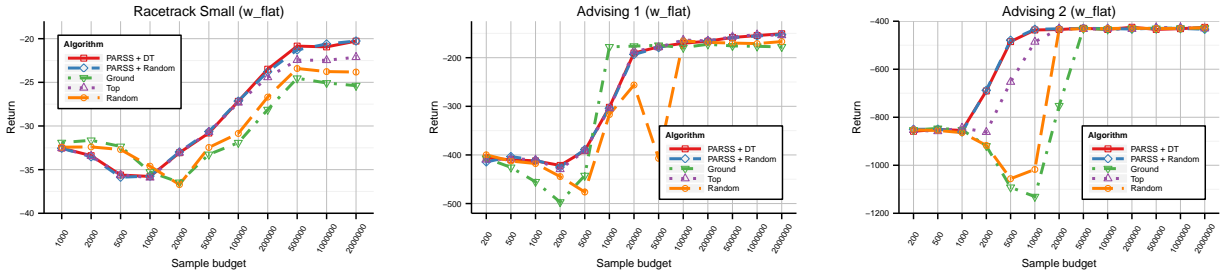


Figure 4: Selected results with best parameters according to  $AUAC(w_{flat})$ .

from TOP in these domains, indicating that abstraction refinement was not beneficial.

The RANDOM abstraction, which is a compromise between TOP and GROUND, tended to fall in between those two abstractions, although RANDOM performed poorly on Saving for small budgets.

The results with time budgets were qualitatively similar to the sample budget results (Figure 3).

To examine the sensitivity of our results to the choice of anytime performance measure, we also evaluated the algorithms with the parameters that maximized  $AUAC$  with respect to the alternative weight function  $w_{flat}(b_i, b_{i-1}) = b_i - b_{i-1}$ . Weighting with  $w_{flat}$  gives equal weight to all budgets in the range, and thus gives greater emphasis to large budgets compared to  $w_{mag}$ . In the domains shown (Figure 4), the different budget weighting resulted in some qualitative changes to the results, but the relative ranking of the algorithms remained the same. Note that the divergence between PARSS and TOP in Advising 2 is due to noise in parameter selection. There were no noteworthy differences in the domains not shown.

There was no difference in performance between PARSS+DT and PARSS+RAND. It appears that either REFINEDT does not find better refinements than REFINERANDOM, or else the quality of refinements does not matter much to overall performance. We expected the simpler REFINERANDOM mechanism to have an advantage under a time budget, but this was not the case.

To summarize, our main experimental findings were:

- TOP was clearly superior to GROUND overall;
- PARSS equaled the performance of TOP, or surpassed TOP through abstraction refinement;
- The two refinement mechanisms — REFINERANDOM and REFINEDT — had identical performance;
- Relative performance results with time budgets were qualitatively similar to those with sample budgets.

## 7 SUMMARY

We have described an extension of sparse sampling called Progressive Abstraction Refinement for Sparse Sampling (PARSS) that adapts the granularity of its state representation to the available planning budget. PARSS exploits the benefits of coarse, unsound abstractions for small budgets while transitioning smoothly to more accurate abstractions when given a larger budget. We proved that PARSS has the same sample complexity and accuracy guarantees as SS. Our experiments demonstrated that planning with the coarsest abstraction, equivalent to open loop planning, yields strong anytime performance on several benchmark domains, and confirmed that PARSS can improve upon that strong performance as budgets increase.

## Acknowledgements

This research was supported by NSF grant IIS 1320943.

## References

- Barto, A. G., Bradtke, S. J., and Singh, S. P. (1995). Learning to act using real-time dynamic programming. *Artificial Intelligence*, 72(1-2):81–138.
- Browne, C. B., Powley, E., Whitehouse, D., Lucas, S. M., Cowling, P. I., Rohlfshagen, P., Tavener, S., Perez, D., Samothrakis, S., and Colton, S. (2012). A survey of Monte Carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(1):1–43.
- Bubeck, S. and Munos, R. (2010). Open loop optimistic planning. In *Conference on Learning Theory (COLT)*.
- Chapman, D. and Kaelbling, L. P. (1991). Input generalization in delayed reinforcement learning: An algorithm and performance comparisons. In *International Joint Conference on Artificial Intelligence (IJCAI)*.
- Guerin, J. T., Hanna, J. P., Ferland, L., Mattei, N., and Goldsmith, J. (2012). The academic advising planning domain. In *Workshop on the International Planning Competition (WS-IPC) at ICAPS*.
- Hauser, K. (2011). Randomized belief-space replanning in partially-observable continuous spaces. In *Algorithmic Foundations of Robotics IX*, pages 193–209. Springer.
- Hostetler, J., Fern, A., and Dietterich, T. (2014). State aggregation in Monte Carlo tree search. In *AAAI Conference on Artificial Intelligence*.
- Jiang, N., Singh, S., and Lewis, R. (2014). Improving UCT planning via approximate homomorphisms. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*.
- Kearns, M., Mansour, Y., and Ng, A. Y. (2002). A sparse sampling algorithm for near-optimal planning in large Markov decision processes. *Machine Learning*, 49(2-3):193–208.
- Kocsis, L. and Szepesvári, C. (2006). Bandit based Monte-Carlo planning. In *European Conference on Machine Learning (ECML)*.
- Li, L., Walsh, T. J., and Littman, M. L. (2006). Towards a unified theory of state abstraction for MDPs. In *International Symposium on Artificial Intelligence and Mathematics*.
- McCallum, A. K. (1996). *Reinforcement learning with selective perception and hidden state*. PhD thesis, University of Rochester.
- Moore, A. W. and Atkeson, C. G. (1995). The parti-game algorithm for variable resolution reinforcement learning in multidimensional state-spaces. *Machine Learning*, 21(3):199–233.
- Pinto, J. and Fern, A. (2014). Learning partial policies to speedup MDP tree search. In *Conference on Uncertainty in Artificial Intelligence (UAI)*.
- Van den Broeck, G. and Driessens, K. (2011). Automatic discretization of actions and states in Monte-Carlo tree search. In *ECML/PKDD Workshop on Machine Learning and Data Mining in and around Games*.
- Walsh, T. J., Goschin, S., and Littman, M. L. (2010). Integrating sample-based planning and model-based reinforcement learning. In *AAAI Conference on Artificial Intelligence*.
- Weinstein, A. and Littman, M. L. (2012). Bandit-based planning and learning in continuous-action Markov decision processes. In *International Conference on Automated Planning and Scheduling (ICAPS)*.
- Weinstein, A. and Littman, M. L. (2013). Open-loop planning in large-scale stochastic domains. In *AAAI Conference on Artificial Intelligence*.



---

# Zero-Truncated Poisson Tensor Factorization for Massive Binary Tensors

---

**Changwei Hu\***  
ECE Department  
Duke University  
Durham, NC 27708

**Piyush Rai\***  
ECE Department  
Duke University  
Durham, NC 27708

**Lawrence Carin**  
ECE Department  
Duke University  
Durham, NC 27708

## Abstract

We present a scalable Bayesian model for low-rank factorization of massive tensors with binary observations. The proposed model has the following key properties: (1) in contrast to the models based on the logistic or probit likelihood, using a zero-truncated Poisson likelihood for binary data allows our model to scale up in the number of *ones* in the tensor, which is especially appealing for massive but sparse binary tensors; (2) side-information in form of binary pairwise relationships (e.g., an adjacency network) between objects in any tensor mode can also be leveraged, which can be especially useful in “cold-start” settings; and (3) the model admits simple Bayesian inference via batch, as well as *online* MCMC; the latter allows scaling up even for *dense* binary data (i.e., when the number of ones in the tensor/network is also massive). In addition, non-negative factor matrices in our model provide easy interpretability, and the tensor rank can be inferred from the data. We evaluate our model on several large-scale real-world binary tensors, achieving excellent computational scalability, and also demonstrate its usefulness in leveraging side-information provided in form of mode-network(s).

## 1 INTRODUCTION

With the recent surge in multiway, multirelational, or “tensor” data sets (Nickel et al., 2011; Kang et al., 2012), learning algorithms that can extract useful knowledge from such data are becoming increasingly important. Tensor decomposition methods (Kolda and Bader, 2009) offer an attractive way to accomplish this task. Among tensor data, of particular interest are real-world *binary* tensors, which are now ubiquitous in problems involving social networks, rec-

ommender systems, and knowledge bases, etc. For instance, in a knowledge base, predicate relations defined over the tuples (subjects, objects, verbs) can be represented in form of a binary three-way tensor (Kang et al., 2012).

Usually, real-world binary tensors are massive (each dimension can be very large) but extremely sparse (very few ones in the tensor). For example, in a recommender system, each positive example (e.g., an item selected a set) implicitly creates several negative examples (items *not* chosen). Likewise, in a knowledge base, the validity of one relation automatically implies invalidity of several other relations. In all these settings, the number of negative examples greatly overwhelms the number of positive examples.

Unfortunately, binary tensor factorization methods (Nickel et al., 2011; Xu et al., 2013; Rai et al., 2014), based on probit or logistic likelihood, scale poorly for massive binary tensors because these require evaluating the likelihood/loss-function on *both* ones as well as zeros in the tensor. One possibility is to use heuristics such as *undersampling* the zeros, but such heuristics usually result in less accurate solutions. Another alternative is to use the *squared loss* (Hidasi and Tikk, 2012; Nickel et al., 2012) as the model-fit criterion, which facilitates linear scalability in the number of ones in the tensor. However, such an approach can often lead to suboptimal results (Ermis and Bouchard, 2014) in practice.

It is therefore desirable to have methods that can perform efficient tensor decomposition for such data, ideally with a computational-complexity that depends only on the number of nonzeros (i.e., the ones) in the tensor, rather than the “volume” of the tensor. Motivated by this problem, we present a scalable Bayesian model for the Canonical PARAFAC (CP) tensor decomposition (Kolda and Bader, 2009), with an inference-complexity that scales linearly in the number of ones in the tensor. Our model uses a zero-truncated Poisson likelihood for each binary observation in the tensor; this obviates the evaluation of the likelihoods for the zero entries. At the same time, the significant speed-up is not at the cost of sacrificing on the quality of the solution. As our experimental results show, the proposed like-

---

\*Equal contribution

likelihood model yields comparable or better results to logistic likelihood based models, while being an order of magnitude faster in its running-time on real-world binary tensors. Note that replacing the zero-truncated Poisson by the standard Poisson makes our model also readily applicable for count-valued tensors (Chi and Kolda, 2012); although, in this exposition, we will focus exclusively on binary tensors.

Often, side-information (Acar et al., 2011; Beutel et al., 2014), e.g., pairwise relationships (partially/fully observed), may also be available for objects in some of the tensor dimensions. For example, in addition to a binary tensor representing AUTHORS  $\times$  WORDS  $\times$  VENUES associations, the AUTHOR  $\times$  AUTHOR co-authorship network may be available (at least for some pairs of authors). Such a network may be especially useful in “cold-start” settings where there is no data for some of the entities of a mode in the tensor (e.g., for some authors, there is no data in the tensor), but a network between entities in that mode may be available (See Fig 1 for an illustration). Our model allows leveraging such network(s), without a significant computational overhead, using the zero-truncated Poisson likelihood *also* to model these binary pairwise relationships.

To facilitate efficient fully Bayesian inference, we develop easy-to-implement batch as well as *online* MCMC inference; the latter is especially appealing for handling *dense* binary data, i.e., when the number of ones in the tensor and/or the network is also massive. Another appealing aspect about the model is its interpretability; a Dirichlet prior on the columns of each factor matrix naturally imposes non-negativity. In addition, the rank of decomposition can be inferred from the data.

## 2 CANONICAL PARAFAC (CP) TENSOR DECOMPOSITION

The Canonical PARAFAC (CP) decomposition (Kolda and Bader, 2009) offers a way to express a tensor as a sum of rank-1 tensors. Each rank-1 tensor corresponds to a specific “factor” in the data. More specifically, the goal in CP decomposition is to decompose a tensor  $\mathcal{Y}$  of size  $n_1 \times n_2 \times \dots \times n_K$ , with  $n_k$  denoting the size of  $\mathcal{Y}$  along the  $k^{\text{th}}$  mode (or “way”) of the tensor, into a set of  $K$  factor matrices  $\mathbf{U}^{(1)}, \dots, \mathbf{U}^{(K)}$  where  $\mathbf{U}^{(k)} = [\mathbf{u}_1^{(k)}, \dots, \mathbf{u}_R^{(k)}]$ ,  $k = \{1, \dots, K\}$ , denotes the  $n_k \times R$  factor matrix associated with mode  $k$ .

In its most general form, CP decomposition expresses the tensor  $\mathcal{Y}$  via a weighted sum of  $R$  rank-1 tensors as

$$\mathcal{Y} \sim f\left(\sum_{r=1}^R \lambda_r \mathbf{u}_r^{(1)} \odot \dots \odot \mathbf{u}_r^{(K)}\right) \quad (1)$$

In the above, the form of the link-function  $f$  depends on the type of data being modeled (e.g.,  $f$  can be Gaussian for real-valued, Bernoulli-logistic for binary-valued, Poisson for count-valued tensors). Here  $\lambda_r$  is the weight associated

with the  $r^{\text{th}}$  rank-1 component, the  $n_k \times 1$  column vector  $\mathbf{u}_r^{(k)}$  represents the  $r^{\text{th}}$  latent factor of mode  $k$ , and  $\odot$  denotes vector outer product.

We use subscript  $\mathbf{i} = \{i_1, \dots, i_K\}$  to denote the  $K$ -dimensional index of the  $\mathbf{i}$ -th entry in the tensor  $\mathcal{Y}$ . Using this notation, the  $\mathbf{i}$ -th entry of the tensor  $\mathcal{Y}$  can be written as  $y_{\mathbf{i}} \sim f\left(\sum_{r=1}^R \lambda_r \prod_{k=1}^K u_{i_k r}^{(k)}\right)$ .

## 3 TRUNCATED POISSON TENSOR DECOMPOSITION FOR BINARY DATA

Our focus in this paper is on developing a probabilistic, fully Bayesian method for scalable low-rank decomposition of massive *binary* tensors. As opposed to tensor decomposition models based on the logistic likelihood for binary data (Xu et al., 2013; Rai et al., 2014), which require evaluation of the likelihood for both ones as well as zeros in the tensor, and thus can be computationally infeasible to run on massive binary tensors, our proposed model only requires the likelihood evaluations on the *nonzero* (i.e., the ones) entries in the tensor, and can therefore easily scale to massive binary tensors. Our model is applicable to tensors of any order  $K \geq 2$  (the case  $K = 2$  being a binary matrix).

Our model is based on a decomposition of the form given in Eq. 1; however, instead of using a Bernoulli-logistic link  $f$  to generate each binary observation  $y_{\mathbf{i}}$  in  $\mathcal{Y}$ , we assume an additional layer (Eq. 2) which takes a *latent* count-valued  $y_{\mathbf{i}}$  in  $\mathcal{Y}$  and thresholds this latent count at one to generate the actual *binary*-valued entry  $b_{\mathbf{i}}$  in the observed binary tensor, which we will denote by  $\mathcal{B}$ :

$$b_{\mathbf{i}} = \mathbf{1}(y_{\mathbf{i}} \geq 1) \quad (2)$$

$$\mathcal{Y} \sim \text{Pois}\left(\sum_{r=1}^R \lambda_r \mathbf{u}_r^{(1)} \odot \dots \odot \mathbf{u}_r^{(K)}\right) \quad (3)$$

$$\mathbf{u}_r^{(k)} \sim \text{Dir}(a^{(k)}, \dots, a^{(k)}) \quad (4)$$

$$\lambda_r \sim \text{Gamma}(g_r, \frac{p_r}{1 - p_r}) \quad (5)$$

$$p_r \sim \text{Beta}(c\epsilon, c(1 - \epsilon)) \quad (6)$$

Marginalizing out  $y_{\mathbf{i}}$  from Eq. 2 leads to the following (equivalent) likelihood model

$$b_{\mathbf{i}} \sim \text{Bernoulli}\left(1 - \exp\left(-\sum_{r=1}^R \lambda_r \prod_{k=1}^K u_{i_k r}^{(k)}\right)\right) \quad (7)$$

Note that the thresholding in (2) looks similar to a probit model for binary data (which however thresholds a *normal* at zero); however, the probit model (just like the logistic model) also needs to evaluate the likelihood at the zeros, and can therefore be slow on massive binary data with lots of zeros. Likelihood models of the form (Eq. 7) have previously also been considered in work on statistical models of undirected networks (Morup et al., 2011; Zhou, 2015).

Interestingly, the form of the likelihood in 7 also resembles the complementary log-log function Collett (2002); Piegorsch (1992), which is known to be a better model for imbalanced binary data than the logistic or probit likelihood, making it ideal for handling sparse binary tensors.

The conditional posterior of the latent count  $y_i$  is given by

$$y_i | b_i, \lambda, \{u_{i_k r}^{(k)}\}_{k=1}^K \sim b_i \cdot \text{Pois}_+(\sum_{r=1}^R \lambda_r \prod_{k=1}^K u_{i_k r}^{(k)}) \quad (8)$$

where  $\text{Pois}_+(\cdot)$  is zero truncated Poisson distribution. Eq. (8) suggests that if  $b_i = 0$ , then  $y_i = 0$  almost surely with probability one, which can lead to significant computational savings, if the tensor has a large number of zeros. In addition, our model also enables leveraging a reparameterization (Section 3.2) of the Poisson distribution in terms of a multinomial, which allows us to obtain very simple Gibbs-sampling updates for the model parameters.

Note that the Dirichlet prior on the latent factors  $\mathbf{u}_r^{(k)}$  naturally imposes non-negativity constraints (Chi and Kolda, 2012) on the factor matrices  $\mathbf{U}^{(1)}, \dots, \mathbf{U}^{(K)}$ . Moreover, since the columns  $\mathbf{u}_r^{(k)}$  of these factor matrices sums to 1, each  $\mathbf{u}_r^{(k)}$  can also be interpreted as a *distribution* (e.g., a “topic”) over the  $n_k$  entities in mode  $k$ . Furthermore, the gamma-beta hierarchical construction (Zhou et al., 2012) of  $\lambda_r$  (Eq 5 and 6) allows inferring the rank of the tensor by setting an upper bound  $R$  on the number of factors and inferring the appropriate number of factors by shrinking the coefficients  $\lambda_r$ ’s to close to zero for the irrelevant factors. These aspects make our model interpretable as well as provide it the ability to do model selection (i.e., inferring the rank), in addition to being computationally efficient by focusing the computations only on the nonzero entries in the tensor  $\mathcal{B}$ .

### 3.1 LEVERAGING MODE NETWORKS

Often, in addition to the binary tensor  $\mathcal{B}$ , *pairwise* relationships between entities in one or more tensor modes may be available in form of a symmetric binary network or an undirected graph. Leveraging such forms of side-information can be beneficial for tensor decomposition, especially if the amount of missing data in the main tensor  $\mathcal{B}$  is very high (Acar et al., 2011; Beutel et al., 2014; Rai et al., 2015), and, even more importantly, in “cold-start” settings, where there is no data in the tensor for entities along some of the tensor mode(s), as shown in Fig 1. In the absence of any side-information, the posterior distribution of the latent factors  $\mathbf{u}_r^{(k)}$  of such entities in that tensor mode would be the same as the prior (i.e., just a random draw). Leveraging the side-information (e.g., a network) helps avoid this.

For entities of the  $k$ -th mode of tensor  $\mathcal{B}$ , we assume a symmetric binary network  $\mathbf{A}^{(k)} \in \{0, 1\}^{n_k \times n_k}$ , where  $A_{i_k j_k}^{(k)}$

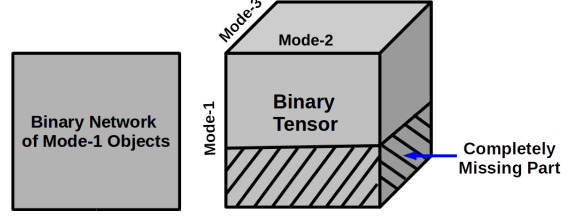


Figure 1: Binary tensor with an associated binary network between objects in mode-1 of the tensor (in general, network for other modes may also be available). In the “cold-start” setting as shown above, data along some of the tensor dimensions will be completely missing

denotes the relationship between mode- $k$  entities  $i_k$  and  $j_k$ .

Just like our tensor decomposition model, we model the mode- $k$  network  $\mathbf{A}^{(k)}$  as a weighted sum of rank-1 symmetric matrices, with a similar likelihood model as we use for the tensor observations. In particular, we assume a latent count  $X_{i_k j_k}^{(k)}$  for each binary entry  $A_{i_k j_k}^{(k)}$ , and threshold it at one to generate  $A_{i_k j_k}^{(k)}$

$$A_{i_k j_k}^{(k)} = \mathbf{1}(X_{i_k j_k}^{(k)} \geq 1) \quad (9)$$

$$\mathbf{X}^{(k)} \sim \text{Pois}(\sum_{r=1}^R \beta_r \cdot \mathbf{u}_r^{(k)} \odot \mathbf{u}_r^{(k)}) \quad (10)$$

$$\beta_r \sim \text{Gamma}(f_r, \frac{h_r}{1 - h_r}) \quad (11)$$

$$h_r \sim \text{Beta}(d\alpha, d(1 - \alpha)) \quad (12)$$

Note that since  $\mathbf{A}^{(k)}$  is symmetric, only the upper (or lower) triangular portion needs to be considered, and moreover, just like in the case of the tensor  $\mathcal{B}$ , due to the truncated Poisson construction, the likelihood at only the nonzero entries needs to be evaluated for this part as well.

### 3.2 REPARAMETERIZED POISSON DRAWS

To simplify posterior inference (Section 4), we make use of two re-parameterizations of a Poisson draw (Zhou et al., 2012). The first parameterization is to express each latent count variable  $y_i$  and  $X_{i_k j_k}^{(k)}$  as a sum of another set of  $R$  latent counts  $\{\tilde{y}_{i r}\}_{r=1}^R$  and  $\{\tilde{X}_{i_k j_k r}^{(k)}\}_{r=1}^R$ , respectively

$$y_i = \sum_{r=1}^R \tilde{y}_{i r}, \quad \tilde{y}_{i r} \sim \text{Pois}(\lambda_r \prod_{k=1}^K u_{i_k r}^{(k)}) \quad (13)$$

$$X_{i_k j_k}^{(k)} = \sum_{r=1}^R \tilde{X}_{i_k j_k r}^{(k)}, \quad \tilde{X}_{i_k j_k r}^{(k)} \sim \text{Pois}(\beta_r u_{i_k r}^{(k)} u_{j_k r}^{(k)}) \quad (14)$$

The second parameterization assumes that the latent counts  $\{\tilde{y}_{i r}\}$  and  $\{\tilde{X}_{i_k j_k r}^{(k)}\}$  are drawn from a multinomial

$$\begin{aligned} \tilde{y}_{i 1}, \dots, \tilde{y}_{i R} &\sim \text{Mult}(y_i; \zeta_{i 1}, \dots, \zeta_{i R}) \\ \zeta_{i r} &= \frac{\lambda_r \prod_{k=1}^K u_{i_k r}^{(k)}}{\sum_{r=1}^R \lambda_r \prod_{k=1}^K u_{i_k r}^{(k)}} \end{aligned} \quad (15)$$

$$\begin{aligned} \tilde{X}_{i_k j_k 1}^{(k)}, \dots, \tilde{X}_{i_k j_k R}^{(k)} &\sim \text{Mult}(X_{i_k j_k}^{(k)}; \kappa_{i_k j_k 1}^{(k)}, \dots, \kappa_{i_k j_k R}^{(k)}) \\ \kappa_{i_k j_k r}^{(k)} &= \frac{\beta_r u_{i_k r}^{(k)} u_{j_k r}^{(k)}}{\sum_{r=1}^R \beta_r u_{i_k r}^{(k)} u_{j_k r}^{(k)}} \end{aligned} \quad (16)$$

As we show in Section 4, these parameterizations enable us to exploit the gamma-Poisson as well as the Dirichlet-multinomial conjugacy to derive simple, closed-form Gibbs sampling updates for the model parameters.

## 4 INFERENCE

Exact inference in the model is intractable and we resort to Markov Chain Monte Carlo (MCMC) (Andrieu et al., 2003) inference. In particular, the reparameterization discussed in Section 3.2 allows us to derive simple Gibbs sampling updates for all the latent variables, except for the latent counts  $y_i$ , which are drawn from a truncated Poisson distribution via rejection sampling. As discussed earlier, the computational-cost for our inference method scales linearly w.r.t. the number of ones in the tensor (plus the number of nonzeros in the network, if side-information is used). This makes our method an order of magnitude faster than models based on logistic or probit likelihood for binary data (Rai et al., 2014; Xu et al., 2013), without sacrificing on the quality of the results. The relative speed-up depends on the ratio of total volume of the tensor to the number of ones, which is given by  $(\prod_{k=1}^K n_k) / \text{nnz}(\mathcal{B})$ ; here  $\text{nnz}(\mathcal{B})$  denotes the number of nonzeros in the tensor.

In this section, we present both batch MCMC (Section 4.1) as well as an online MCMC (Section 4.2) method for inference in our model. The online MCMC algorithm is based on the idea of Bayesian Conditional Density Filtering (CDF) (Guhaniyogi et al., 2014), and can lead to further speed-ups over the batch MCMC if the number of nonzeros in the tensor is also massive. The CDF algorithm provides an efficient way to perform online MCMC sampling using surrogate conditional sufficient statistics (Guhaniyogi et al., 2014).

For both batch MCMC and CDF based online MCMC, we provide the update equations, with and without the side-information, i.e., the mode network(s). For what follows, we define four quantities:  $s_{j,r}^{(k)} = \sum_{i:i_k=j} \tilde{y}_{i_r}$ ,  $s_r = \sum_i \tilde{y}_{i_r}$ ,  $v_{i_k,r} = \sum_{j_k} \tilde{X}_{i_k j_k r}^{(k)}$  and  $v_r = \sum_{i_k} \sum_{j_k} \tilde{X}_{i_k j_k r}^{(k)}$ , which denote aggregates computed using the latent counts  $\tilde{y}_{i_r}$  and  $\tilde{X}_{i_k j_k r}^{(k)}$ . These quantities will be used at various places in the description of the inference algorithms that we present here.

### 4.1 BATCH MCMC INFERENCE

#### 4.1.1 Tensor without Mode Network(s)

**Sampling  $y_i$ :** For each observation  $b_i$  in the tensor, the latent count  $y_i$  is sampled as

$$y_i \sim b_i \cdot \text{Pois}_+(\sum_{r=1}^R \lambda_r \prod_{k=1}^K u_{i_k r}^{(k)}) \quad (17)$$

where  $\text{Pois}_+(\cdot)$  is zero truncated Poisson distribution. Eq. (17) suggests that if  $b_i = 0$ , then  $y_i = 0$  almost surely; and if  $b_i = 1$ , then  $y_i \sim \text{Pois}_+(\sum_{r=1}^R \lambda_r \prod_{k=1}^K u_{i_k r}^{(k)})$ . Therefore the  $y_i$ 's only need to be sampled for the nonzero  $b_i$ 's. **Sampling  $\tilde{y}_{i_r}$ :** The latent counts  $\{\tilde{y}_{i_r}\}$  are sampled from a multinomial as Eq. (15). Note that this also needs to be done only for the nonzero  $b_i$ 's.

**Sampling  $\mathbf{u}_r^{(k)}$ :** The columns of each factor matrix have a Dirichlet posterior, and are sampled as

$$\mathbf{u}_r^{(k)} \sim \text{Dir}(a^{(k)} + s_{1,r}^{(k)}, a^{(k)} + s_{2,r}^{(k)}, \dots, a^{(k)} + s_{n_k,r}^{(k)}) \quad (18)$$

**Sampling  $p_r$ :** Using the fact that  $s_r = \sum_i \tilde{y}_{i_r}$  and marginalizing over the  $u_{i_k r}^{(k)}$ 's in (13), we have  $s_r \sim \text{Pois}(\lambda_r)$ . Using this, along with (5), we can express  $s_r$  using a negative binomial distribution, i.e.,  $s_r \sim \text{NB}(g_r, p_r)$ . Due to the conjugacy between negative binomial and beta, we can then sample  $p_r$  as

$$p_r \sim \text{Beta}(c\epsilon + s_r, c(1 - \epsilon) + g_r) \quad (19)$$

**Sampling  $\lambda_r$ :** Again using the fact that  $s_r \sim \text{Pois}(\lambda_r)$  and (5), we have

$$\lambda_r \sim \text{Gamma}(g_r + s_r, p_r) \quad (20)$$

As can be observed, when updating  $\mathbf{u}_r^{(k)}$ ,  $p_r$  and  $\lambda_r$ , the latent counts  $y_i$ 's and  $\tilde{y}_{i_r}$  corresponding to zero entries in  $\mathcal{B}$  are all equal to zero, and have no contribution to sufficient statistics  $s_{j,r}^{(k)}$  and  $s_r$ . Therefore, only the nonzero entries in tensor need to be considered in the computations.

#### 4.1.2 Tensor with Mode Network(s)

In the presence of mode network(s), the update equations for the latent variables  $p_r$ ,  $\lambda_r$ ,  $\tilde{y}_{i_r}$  and  $y_i$ , that are associated solely with the binary tensor  $\mathcal{B}$ , remain unchanged, and can be sampled as described in Section 4.1.1. We however need to sample the additional latent variables associated with mode- $k$  network  $\mathbf{A}^{(k)}$ , and the latent factors  $\mathbf{u}_r^{(k)}$  of mode- $k$  that are shared by the binary tensor  $\mathcal{B}$  as well as the mode- $k$  network.

**Sampling  $X_{i_k j_k}^{(k)}$ :** The latent counts  $X_{i_k j_k}^{(k)}$  are sampled as

$$X_{i_k j_k}^{(k)} \sim A_{i_k j_k}^{(k)} \cdot \text{Pois}_+(\sum_{r=1}^R \beta_r u_{i_k r}^{(k)} u_{j_k r}^{(k)}) \quad (21)$$

This only needs to be done for the nonzero entries in  $\mathbf{A}^{(k)}$ .

**Sampling  $\tilde{X}_{i_k j_k r}^{(k)}$ :** The latent counts  $\tilde{X}_{i_k j_k r}^{(k)}$  are sampled from a multinomial as equation (16). This also only needs to be done for the nonzero entries in  $\mathbf{A}^{(k)}$ .

**Sampling  $\mathbf{u}_r^{(k)}$ :** The columns of each factor matrix have a Dirichlet posterior, and are sampled as

$$\mathbf{u}_r^{(k)} \sim \text{Dir}(a^{(k)} + s_{1,r}^{(k)} + v_{1,r}, \dots, a^{(k)} + s_{n_k,r}^{(k)} + v_{n_k,r}) \quad (22)$$

Note that in the absence of the mode- $k$  network, the terms  $v_{\cdot,r}$  go away and Eq. 22 simply reduces to Eq. 18.

**Sampling  $h_r$ :**  $h_r \sim \text{Beta}(d\alpha + v_r, d(1 - \alpha) + f_r)$ .

**Sampling  $\beta_r$ :**  $\beta_r \sim \text{Gamma}(f_r + v_r, h_r)$ .

### 4.1.3 Per-iteration time-complexity

For the binary tensor  $\mathcal{B}$ , computing each  $\zeta_{i_r}$  (Eq. 15) takes  $\mathcal{O}(K)$  time and therefore computing all the  $\{\zeta_{i_r}\}$  takes  $\mathcal{O}(\text{nnz}(\mathcal{B})RK)$  time. Likewise, for the binary mode- $k$  network  $\mathbf{A}^{(k)}$ , computing all the  $\{\kappa_{i_k j_k r}^{(k)}\}$  (Eq. 16) takes  $\mathcal{O}(\text{nnz}(\mathbf{A}^{(k)})R)$  time. These are the most dominant computations in each iteration of our MCMC procedure; updating each  $\mathbf{u}_r^{(k)}$  takes  $\mathcal{O}(n_k)$  time and updating  $\{p_r, h_r\}_{r=1}^R$  and  $\{\lambda_r, \beta_r\}_{r=1}^R$  takes  $\mathcal{O}(R)$  time each. Therefore, the per-iteration time-complexity of our batch MCMC method is  $\mathcal{O}(\text{nnz}(\mathcal{B})RK + \text{nnz}(\mathbf{A}^{(k)})R)$ . The linear dependence on  $\text{nnz}(\mathcal{B})$ ,  $\text{nnz}(\mathbf{A}^{(k)})$ ,  $R$  and  $K$  suggests that even massive, sparse binary tensors and mode network(s) can be handled easily even by our simple batch MCMC implementation. Also note that our model scales linearly even w.r.t.  $R$ , unlike most other methods (Ermis and Bouchard, 2014; Rai et al., 2014) that have *quadratic* dependence on  $R$ .

The above computations can be further accelerated using a distributed/multi-core setting; we leave this for future work. In Section 4.2, however, we present an *online* MCMC method based on the idea of Bayesian Conditional Density Filtering (Guhaniyogi et al., 2014), which leads to further speed-ups, even in single-machine settings.

## 4.2 ONLINE MCMC INFERENCE

We develop an efficient online MCMC sampler for the model, leveraging ideas from the Conditional Density Filtering (CDF) Guhaniyogi et al. (2014). The CDF algorithm for our model selects a minibatch of the tensor (and mode network, if the side-information is available) entries at each iteration, samples the model parameters from the posterior, and updates the sufficient statistics  $s_{j,r}^{(k)}$ ,  $s_r$ ,  $v_{i_k,r}$  and  $v_r$  using the data from the current minibatch.

### 4.2.1 Tensor without Mode Network(s)

We first provide the update equations for the case when there is no side-information (mode network). Denote  $I_t$  as indices of entries of tensor  $\mathcal{B}$  from the minibatch selected at iteration  $t$ . The CDF algorithm at iteration  $t$  proceeds as:

**Sampling  $y_i$ :** For all  $i \in I_t$ , sample  $y_i$  according to equation (17); like in the batch MCMC case, the sampling only

needs to be done for the nonzero  $b_i$ 's.

**Sampling  $\tilde{y}_{i_r}$ :** For all  $i \in I_t$ , sample the latent counts  $\tilde{y}_{i_r}(i \in I_t)$  using (15), again only for the nonzero  $b_i$ 's.

**Updating the conditional sufficient statistics:** Update the conditional sufficient statistics  $s_{j,r}^{(k)}$  as  $s_{j,r}^{(k,t)} = s_{j,r}^{(k,t-1)} + \sum_{i \in I_t: i_k=j} \tilde{y}_{i_r}$  and update  $s_r$  as  $s_r^{(t)} = s_r^{(t-1)} + \sum_{i \in I_t} \tilde{y}_{i,r}$ . These updates basically add to the old sufficient statistics, the contributions from the data in the current minibatch. In practice, we also *reweight* these sufficient statistics by the ratio of the total number of ones in  $\mathcal{B}$  and the minibatch size, so that they represent the average statistics over the entire tensor. This reweighting is akin to the way average gradients are computed in stochastic variational inference methods (Hoffman et al., 2013).

**Updating  $\mathbf{u}_r^{(k)}$ ,  $p_r$ ,  $\lambda_r$ :** Using the following conditionals, draw  $M$  samples  $\{\mathbf{u}_r^{(k,m)}, p_r^{(m)}, \lambda_r^{(m)}\}_{m=1}^M$

$$\mathbf{u}_r^{(k)} \sim \text{Dir}(a^{(k)} + s_{1,r}^{(k,t)}, \dots, a^{(k)} + s_{n_k,r}^{(k,t)}) \quad (23)$$

$$p_r \sim \text{Beta}(c\epsilon + s_r^{(t)}, c(1 - \epsilon) + g_r) \quad (24)$$

$$\lambda_r \sim \text{Gamma}(g_r + s_r^{(t)}, p_r) \quad (25)$$

and either store the sample averages of  $\mathbf{u}_r^{(k)}$ ,  $p_r$ , and  $\lambda_r$ , or their analytic means to use for the next CDF iteration (Guhaniyogi et al., 2014).

### 4.2.2 Tensor with Mode Network(s)

For all the latent variables associated solely with the tensor  $\mathcal{B}$ , the sampling equations for the CDF algorithm in the presence of mode network(s) remain unchanged as the previous case with no network. In the presence of the mode network, the additional latent variables include the sufficient statistics  $v_{i_k,r}$  and  $v_r$ , and these need to be updated in each CDF iteration.

Denote  $J_t$  as indices of entries selected from the mode- $k$  network  $\mathbf{A}^{(k)}$  in iteration  $t$ . The update equations for the latent variables that depend on  $\mathbf{A}^{(k)}$  are as follows:

**Sampling  $X_{i_k j_k}$ :** For  $(i_k, j_k) \in J_t$ , latent count  $X_{i_k j_k}$  is sampled using Eq. (21).

**Sampling  $\tilde{X}_{i_k j_k r}$ :** For  $(i_k, j_k) \in J_t$ , latent counts  $\tilde{X}_{i_k j_k r}$  are sampled from a multinomial using Eq. (16).

**Updating the conditional sufficient statistics:** Update the sufficient statistics associated with the mode- $k$  network as  $v_{i_k,r}^{(t)} = v_{i_k,r}^{(t-1)} + \sum_{j_k, (i_k, j_k) \in J_t} \tilde{X}_{i_k j_k r}^{(t)}$  and  $v_r^{(t)} = v_r^{(t-1)} + \sum_{i_k} \sum_{j_k, (i_k, j_k) \in J_t} \tilde{X}_{i_k j_k r}^{(t)}$ . Just like the way we update the tensor sufficient statistics  $s_{j,r}^{(k)}$  and  $s_r$ , we reweight these mode- $k$  sufficient statistics by the ratio of the total number of ones in  $\mathbf{A}^{(k)}$  and the minibatch size, so that they represent the average statistics over the entire mode- $k$  network.

**Updating  $\mathbf{u}_r^{(k)}$ ,  $h_r$ ,  $\beta_r$ :** Using the following condition-

als, draw  $M$  samples  $\{\mathbf{u}_r^{(k,m)}, h_r^{(m)}, \beta_r^{(m)}\}_{m=1}^M$ . We draw  $\mathbf{u}_r^{(k)} \sim \text{Dir}(a^{(k)} + s_{1,r}^{(k,t)} + v_{1,r}^{(t)}, \dots, a^{(k)} + s_{n_k,r}^{(k,t)} + v_{n_k,r}^{(t)})$ , and  $h_r$  and  $\beta_r$  as

$$\begin{aligned} h_r &\sim \text{Beta}(d\alpha + v_r^{(t)}, d(1 - \alpha) + f_r) \\ \beta_r &\sim \text{Gamma}(f_r + v_r^{(t)}, h_r) \end{aligned} \quad (26)$$

and either store the sample averages of  $\mathbf{u}_r^{(k)}$ ,  $h_r$ ,  $\beta_r$ , or their analytic means to use for the next CDF iteration.

### 4.2.3 Per-iteration time-complexity

The per-iteration time-complexity of the CDF based online MCMC is linear in the number of nonzeros in each minibatch (as opposed to the batch MCMC where it depends on the number of nonzeros in the *entire* tensor and network). Therefore the online MCMC is attractive for *dense* binary data, where the number of nonzeros in the tensor/network is also massive; using a big-enough minibatch size (that fits in the main memory and/or can be processed in each iteration in a reasonable amount of time), the online MCMC inference allows applying our model on such dense binary data as well, which may potentially have several billions of nonzero entries.

## 5 RELATED WORK

With the increasing prevalence of structured databases, social networks, and (multi)relational data, tensor decomposition methods are becoming increasingly popular for extracting knowledge and doing predictive analytics on such data (Bordes et al., 2011; Nickel et al., 2012; Kang et al., 2012). As the size of these data sets continues to grow, there has been a pressing need to design tensor factorization methods that can scale to massive tensor data.

For low-rank factorization of *binary* tensors, methods based on logistic and probit likelihood for the binary data have been proposed (Jenatton et al., 2012; London et al., 2013; Rai et al., 2014; Xu et al., 2013). However, these methods are not suited for massive binary tensors where the number of observations (which mostly consist of zeros, if the tensor is also sparse) could easily be millions or even billions (Inah et al., 2015). As a heuristic, these methods rely on subsampling (Rai et al., 2014) or partitioning the tensor (Zhe et al., 2015), to select a manageable number entries before performing the tensor decomposition, or alternatively going for a distributed setting (Zhe et al., 2013).

In the context of tensor factorization, to the best of our knowledge, the only method (and one that is closest in spirit to our work) that scales linearly w.r.t. the number of ones in the tensor is (Ermis and Bouchard, 2014). Their work explored quadratic loss (and its variations) as a surrogate to the logistic loss and proposed a method (Quad-Approx) with a per-iteration complexity  $\mathcal{O}(\text{nnz}(\mathcal{B})R + R^2 \sum_{k=1}^K n_k)$ . Note that its dependence on

$R$  is quadratic as opposed to our method which is also linear in  $R$ . They also proposed variations based on piecewise quadratic approximations; however, as reported in their experiments (Ermis and Bouchard, 2014), these variations were found to be about twice as slow than their basic Quad-Approx method (Ermis and Bouchard, 2014). Moreover, their methods (and the various other methods discussed in this section) have several other key differences from our proposed model: (1) our model naturally imposes non-negativity on the factor matrices; (2)  $R$  can be inferred from data; (3) our method provides a fully Bayesian treatment; (4) in contrast to their method, which operates in a batch setting, the online MCMC inference allows our model to scale to even bigger problems, where the number of nonzeros could also be massive; and (5) our model also allows incorporating (fully or partially observed) mode-networks as a rich source of side-information.

In another recent work (Zhou, 2015), a similar zero-truncated Poisson construction, as ours, was proposed for *edge-partitioning* based network clustering, allowing the proposed model to scale in terms of the number of edges in the network. Our model, on the other hand, is more general and can be applied to multiway binary tensor data, with an optionally available binary network as a potential source of side-information. Moreover, the Dirichlet prior on the factor matrices, its reparametrizations (Section 3.2), and the online MCMC inference lead to a highly scalable framework for tensor decomposition with side-information.

Another line of work on scaling up tensor factorization methods involves developing distributed and parallel methods (Kang et al., 2012; Inah et al., 2015; Papalexakis et al., 2012; Beutel et al., 2014). Most of these methods, however, have one or more of the following limitations: (1) these methods lack a proper generative model of the data, which is simply assumed to be real-valued and the optimization objective is based on minimizing the Frobenius norm of the tensor reconstruction error, which may not be suitable for binary data; (2) these methods usually assume a parallel or distributed setting, and therefore are not feasible to run on a single machine; (3) missing data cannot be easily handled/predicted; and (4) the rank of the decomposition needs to be chosen via cross-validation.

Leveraging sources of side-information for tensor factorization has also been gaining a lot of attention recently. However, most of these methods cannot scale easily to massive tensors (Acar et al., 2011; Rai et al., 2015), or have to rely on parallel or distributed computing infrastructures (Beutel et al., 2014). In contrast, our model, by the virtue of its scalability that only depends on the number of nonzero entries in the tensor and/or the mode network, easily allows it to scale to massive binary tensors, with or without mode-network based side-information.

## 6 EXPERIMENTS

We report experimental results for our model on a wide range of real-world binary tensors (with and without mode-network based side-information), and compare it with several baselines for binary tensor factorization. We use the following data sets for our experiments:

- **Kinship:** This is a binary tensor of size  $104 \times 104 \times 26$ , representing 26 types of relationships between 104 members of a tribe (Nickel et al., 2011). The tensor has about 3.8% nonzeros.
- **UMLS:** This is a binary tensor of size  $135 \times 135 \times 49$  representing 56 types of verb relations between 135 high-level concepts (Nickel et al., 2011). The tensor has about 0.8% nonzeros.
- **MovieLens:** This is a binary *matrix* (two-way tensor) of size  $943 \times 1682$  representing the binary ratings (thumbs-up or thumbs-down) by 943 users on 1682 movies<sup>1</sup>. This data set has a total of 100,000 ones.
- **DBLP:** This is a binary tensor of size  $10,000 \times 200 \times 10,000$  representing (author-conference-keyword) relations (Zhe et al., 2015). This tensor has only about 0.001% nonzeros, and is an ideal example of a massive but sparse binary tensor.
- **Scholars:** This is a binary tensor of size  $2370 \times 8663 \times 4066$ , constructed from a database of research paper abstracts published by researchers at Duke University; the three tensor modes correspond to authors, words, and publication venues, respectively. Just like the DBLP data, this tensor is also massive but extremely sparse with only about 0.002% nonzeros. In addition, the co-authorship network (i.e., who has written papers with whom) is also available, which we use as a source of side-information, and use this network to experiment with the *cold-start* setting (i.e., when the main tensor has no information about some authors).
- **Facebook:** The Facebook data is a binary tensor of size  $63731 \times 63730 \times 1837$  with the three modes representing wall-owner, poster, and days (Papalexakis et al., 2013). This tensor has only 737498 nonzeros. In addition to the binary tensor, the social network (friendship-links) between users is also given in form of a symmetric binary matrix of size  $63731 \times 63731$ , which has 1634180 nonzeros. We use the network to experiment with the cold-start setting.

We use all the 6 data sets for the tensor completion experiments (Section 6.1). We also use the Scholars and Facebook data in the cold-start setting, where we experiment on the tensor completion task, leveraging the mode-network based side-information (Section 6.4).

<sup>1</sup><http://grouplens.org/datasets/movielens/>

The set of experiments we perform includes: (1) binary tensor completion (Section 6.1) using only the tensor data; (2) scalability behavior of our model (both batch as well as online MCMC) in terms of tensor completion accuracy vs run-time (Section 6.2); we compare our model with Bayesian CP based on logistic-likelihood (Rai et al., 2014); (3) a qualitative analysis of our results using a *multiway* topic modeling experiment (Section 6.3) on the Scholars data, with the entities being authors, words, and publication venues; and (4) leveraging the mode network for tensor completion in the cold-start setting (Section 6.4); for this experiment, we also demonstrate how leveraging the network leads to improved qualitative results in the multi-way topic modeling problem.

In the experiments, we refer to our model as **ZTP-CP** (Zero-Truncated Poisson based CP decomposition). We compare **ZTP-CP** (using both batch MCMC as well as online MCMC inference) with the following baselines: (1) the quadratic loss minimization (**Quad-App**) proposed in (Ermis and Bouchard, 2014); (2) the refined piecewise quadratic approximation algorithm (**PW-QuadApp**) (Ermis and Bouchard, 2014); and (3) **Bayesian CP** decomposition based on logistic likelihood for binary data (Rai et al., 2014).

**Experimental settings:** All experiments are done on a standard desktop computer with Intel i7 3.4GHz processor and 24GB RAM. Unless specified otherwise, the MCMC inference was run for 1000 iterations with 500 burn-in iterations. The online MCMC algorithm was also run for the same number of iterations, with minibatch size equal to one-tenth of the number of nonzeros in the training data. For all the data sets, except Scholars and Facebook, we use  $R = 20$  (also note that our model has the ability to prune the unnecessary factors by shrinking the corresponding  $\lambda_r$  to zero). For Scholars and Facebook data, we set  $R = 100$ . The hyperparameters  $g_r, f_r$  were set to 0.1, and  $\epsilon$  and  $\alpha$  are set to  $1/R$ , which worked well in our experiments.

### 6.1 TENSOR COMPLETION

In Table 1, we report the results on the tensor completion task (in terms of the AUC-ROC - the area under the ROC curve). For this experiment, although available, we do not use the mode network for the Scholars and the Facebook data; only the binary tensor is used (the results when also using the network are reported in Section 6.4). For each data set, we randomly select 90% of the tensor observations as the training data and evaluate each model on the remaining 10% observations used as the held-out data.

Since the code for **Quad-App** and **PW-QuadApp** baselines (both proposed in (Ermis and Bouchard, 2014)) is not publicly available, we are only able to report the results for the Kinship, UMLS, and MovieLens data set (using the results reported in (Ermis and Bouchard, 2014)). For Bayesian CP (Rai et al., 2014), we use the code provided

Table 1: Tensor completion accuracies in terms of AUC-ROC scores. Results are averaged over 10 splits of training and test data. Note: (1) Bayesian CP was infeasible to run on the Scholars and Facebook data; (2) Due to the lack of publicly available code for **Quad-App** and **PQ-QuadApp**, we only report its results on Kinship, UMLS, and MovieLens data (results taken from (Ermis and Bouchard, 2014)).

|  | Kinship       | UMLS          | MovieLens     | DBLP          | Scholars      | Facebook      |
|--|---------------|---------------|---------------|---------------|---------------|---------------|
| <b>Quad-App (Ermis and Bouchard, 2014)</b>     | 0.8193        | 0.8205        | 0.8511        | -             | -             | -             |
| <b>PW-QuadApp (Ermis and Bouchard, 2014)</b>   | 0.9213        | 0.9387        | 0.9490        | -             | -             | -             |
| <b>Bayesian-Logistic-CP (Rai et al., 2014)</b> | <b>0.9865</b> | <b>0.9965</b> | 0.9799        | 0.9307        | -             | -             |
| <b>ZTP-CP (Batch MCMC)</b>                     | 0.9674        | 0.9938        | <b>0.9895</b> | <b>0.9759</b> | <b>0.9959</b> | 0.9830        |
| <b>ZTP-CP (Online MCMC)</b>                    | 0.9628        | 0.9936        | 0.9841        | 0.9743        | 0.9958        | <b>0.9844</b> |

by the authors. Moreover, the Bayesian CP baseline was found infeasible to run on the Scholars and Facebook data (both of which are massive tensors), so we are unable to report those results. For fairness, on Kinship, UMLS, and MovieLens data, we use the same experimental settings for all the methods as used by (Ermis and Bouchard, 2014).

As shown in Table 1, our model outperforms **Quad-App** and **PW-QuadApp** in terms of the tensor-completion accuracies, and performs comparably or better than **Bayesian CP**, while being an order of magnitude faster (Section 6.2 shows the results on running times).

## 6.2 SCALABILITY

We next compare our model with Bayesian CP (Rai et al., 2014) in terms of the running times vs tensor completion accuracy on Kinship and UMLS data sets. As shown in Fig. 2 (top-row), our model (batch as well as online MCMC) runs/converges an order of magnitude faster than Bayesian CP in terms of running time. On Scholars and Facebook, since Bayesian CP was infeasible to run, we are only able to show the results (Fig. 2, bottom-row) for our model, with batch MCMC and online MCMC inference. On all the data sets, the online MCMC runs/converges faster than the batch MCMC.

We would like to note that, although the model proposed in (Ermis and Bouchard, 2014) also scales linearly <sup>2</sup> in the number of ones in the tensor, the per-iteration time-complexity of our model, which is linear in *both*  $\text{nnz}(\mathcal{B})$  as well as rank  $R$ , is better than the model proposed in (Ermis and Bouchard, 2014) (which has *quadratic* dependence on  $R$ ). Moreover, the tensor completion results of our model (shown in Table 1) on these data sets are better than the ones reported in (Ermis and Bouchard, 2014).

## 6.3 MULTIWAY TOPIC MODELING

We also apply our model for a *multiway* topic modeling task on the Scholars data. The binary tensor represents  $\text{AUTHORS} \times \text{WORDS} \times \text{VENUES}$  relationships. We apply our model (with batch MCMC) and examine the latent factors of each of the three dimensions. Since each factor is drawn from a Dirichlet, it is non-negative and naturally cor-

<sup>2</sup>Although (Ermis and Bouchard, 2014) reported run times on Kinship and UMLS data sets, those number are not directly comparable with our run times reported here (due to possibly different machine configuration, which they do not specify in the paper).

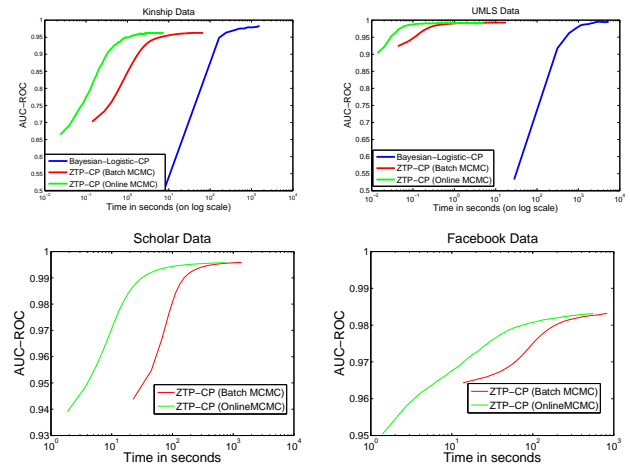


Figure 2: Running time (log-scale) comparison of various methods on Kinship (top left), UMLS (top right), Scholars (bottom left), and Facebook (bottom right) datasets.

responds to a “topic”. In Table 2, after examining the words factor matrix, we show the top-10 words for four of the factors (topics) inferred by our model; these factors seem to represent topics Evolutionary Biology, Medical Imaging, Machine Learning/Signal Processing, and Oncology. For the Machine Learning/Signal Processing topic, we also examine the corresponding topic in the venues factor matrix and show the top-10 venues in that topic (based on their factor scores in that factor). In Fig. 3, we also show the histograms of authors’ department affiliations for each of the four topics and the results make intuitive sense. The results in Table 2 and Fig. 3 demonstrate the usefulness of our model for scalable topic modeling of such multiway data.

## 6.4 LEVERAGING THE MODE NETWORK

Finally, to investigate the usefulness of leveraging the mode network, we experiment with using both the tensor *and* the mode network on Scholars and Facebook data sets. For each data set, we report the AUC-ROC (area under the ROC curve) and AUC-PR (area under the precision-recall curve) on the tensor completion task, with and without network. For both data sets, we experiment with the more challenging cold-start setting. In particular, for the Facebook data, we hold out all the entries of the tensor slices after the first 50,000 wall-owners and predict those entries (using only the rest of the tensor, and using the rest of the tensor as well as the friendship network). We run the experiment with  $R = 20$  and minibatch size of 50,000 for the online



Table 2: For the Scholars data, the most probable words in topics related to evolutionary biology (Evo Bio), medical imaging (Med Imag), machine learning/signal processing(ML/SP) and oncology, and top ranked venues in ML/SP

| EVO BIO      | MED IMAG   | ML/SP         | ONCOLOGY     | TOP VENUES IN ML/SP     |
|--------------|------------|---------------|--------------|-------------------------|
| SPECIES      | IMAGING    | BAYESIAN      | RADIATION    | ICASSP                  |
| SELECTION    | CONTRAST   | ALGORITHM     | RADIOTHERAPY | JASA                    |
| GENETIC      | COMPUTED   | SAMPLING      | STAGE        | ICML                    |
| EVOLUTION    | RESONANCE  | FEATURES      | TUMOR        | IEEE TRANS IMG PROC     |
| POPULATIONS  | DOSE       | PROCESS       | SURVIVAL     | NIPS                    |
| EVOLUTIONARY | TOMOGRAPHY | SPARSE        | LUNG         | COMPU STAT DATA ANALY   |
| GENE         | MAGNETIC   | NONPARAMETRIC | CHEMOTHERAPY | BIOMETRICS              |
| VARIATION    | IMAGE      | GIBBS         | TREATED      | BAYESIAN ANALYSIS       |
| PLANTS       | QUALITY    | PARAMETERS    | TOXICITY     | JMLR                    |
| NATURAL      | DIAGNOSTIC | INFERENCE     | ONCOLOGY     | IEEE TRANS. INF. THEORY |

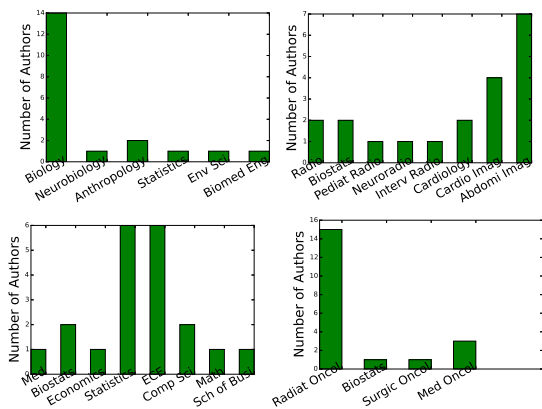


Figure 3: Histogram of the department-affiliations for the top 20 authors in factors related to evolutionary biology (top left), medical imaging (top right), machine learning/signal processing(bottom left) and oncology (bottom right).

MCMC. The results in Table 3 show that using the network leads to better tensor completion accuracies.

We also perform a similar experiment on the Scholars data where we hold out all the entries in tensor slices after the first 1000 authors and predict those entries (using only the rest of the tensor, and using the rest of the tensor as well as the co-authorship network). We run the experiment with  $R = 100$  and minibatch size of 50,000 for the online MCMC. The results shown in Table 3 again demonstrate the benefit of using the network.

Table 3: Cold-start setting

|                        | Facebook      |               | Scholars      |               |
|------------------------|---------------|---------------|---------------|---------------|
|                        | AUC-ROC       | AUC-PR        | AUC-ROC       | AUC-PR        |
| <b>Without network</b> | 0.8897        | 0.6076        | 0.8051        | 0.5763        |
| <b>With network</b>    | <b>0.9075</b> | <b>0.7255</b> | <b>0.8124</b> | <b>0.6450</b> |

In Fig. 4, we show another result demonstrating the benefit of using the co-authorship network for the Scholars data. Note that in the cold-start setting, there is no information in the tensor for the *held-out* authors. Therefore the topics associated with such authors are expected to be roughly *uniformly random*. As shown in Fig. 4 (left column), the set of held-out authors assigned to the topics medical imaging and oncology seem very random and *arbitrary* (we only show the aggregate department-affiliations). Using side-information (in form of the co-authorship network), however, the model sensibly assigns authors who are indeed related to these topics, as shown in right column of Fig. 4.

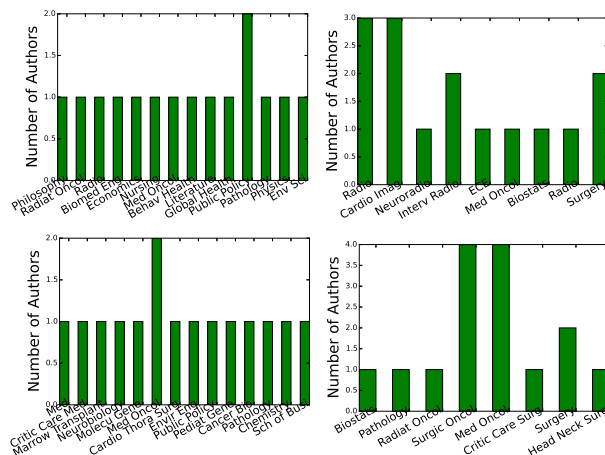


Figure 4: Histogram of the department-affiliations of the top 15 held-out authors associated with the factors of medical imaging (top) and oncology (bottom). The left column is obtained using no co-authorship information, and the right column is obtained using co-authorship information.

## 7 CONCLUSION

We have presented a scalable Bayesian model for binary tensor factorization. In contrast to the models based on probit or logistic likelihood for binary tensor decomposition, the time-complexity of our model depends only in the number of ones in the tensor. This aspect of our model allows it to easily scale up to massive binary tensors. The simplicity of our model also leads to simple batch as well as on-line MCMC inference; the latter allows our model to scale up even when the number of ones could be massive. Our experimental results demonstrate that the model leads to speed-ups of an order of magnitude when compared to binary tensor factorization models based on the logistic likelihood, and also outperforms various other baselines. Our model also gives interpretable results which helps qualitative analysis of results. In addition, the ability to leverage mode networks (fully or partially observed) leads to improved tensor decomposition in cold-start problems.

## Acknowledgments

The research reported here was supported in part by ARO, DARPA, DOE, NGA and ONR.

## References

- Acar, E., Kolda, T. G., and Dunlavy, D. M. (2011). All-at-once optimization for coupled matrix and tensor factorizations. *arXiv preprint arXiv:1105.3422*.
- Andrieu, C., De Freitas, N., Doucet, A., and Jordan, M. I. (2003). An introduction to mcmc for machine learning. *Machine learning*, 50(1-2):5–43.
- Beutel, A., Kumar, A., Papalexakis, E. E., Talukdar, P. P., Faloutsos, C., and Xing, E. P. (2014). Flexifact: Scalable flexible factorization of coupled tensors on hadoop. In *SDM*.
- Bordes, A., Weston, J., Collobert, R., and Bengio, Y. (2011). Learning structured embeddings of knowledge bases. In *AAAI*.
- Chi, E. C. and Kolda, T. G. (2012). On tensors, sparsity, and nonnegative factorizations. *SIAM Journal on Matrix Analysis and Applications*, 33(4):1272–1299.
- Collett, D. (2002). *Modelling binary data*. CRC press.
- Ermis, B. and Bouchard, G. (2014). Iterative splits of quadratic bounds for scalable binary tensor factorization. In *UAI*.
- Guhaniyogi, R., Qamar, S., and Dunson, D. B. (2014). Bayesian conditional density filtering. *arXiv preprint arXiv:1401.3632*.
- Hidasi, B. and Tikk, D. (2012). Fast ALS-based tensor factorization for context-aware recommendation from implicit feedback. In *ECML PKDD*.
- Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. (2013). Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347.
- Inah, J., Papalexakis, E., Kang, U., and Faloutsos, C. (2015). Haten2: Billion-scale tensor decompositions. In *ICDE*. IEEE.
- Jenatton, R., Le Roux, N., Bordes, A., and Obozinski, G. (2012). A latent factor model for highly multi-relational data. In *NIPS*.
- Kang, U., Papalexakis, E., Harpale, A., and Faloutsos, C. (2012). Gigatensor: scaling tensor analysis up by 100 times-algorithms and discoveries. In *KDD*.
- Kolda, T. G. and Bader, B. W. (2009). Tensor decompositions and applications. *SIAM review*, 51(3):455–500.
- London, B., Rekatsinas, T., Huang, B., and Getoor, L. (2013). Multi-relational learning using weighted tensor decomposition with modular loss. *arXiv preprint arXiv:1303.1733*.
- Morup, M., Schmidt, M. N., and Hansen, L. K. (2011). Infinite multiple membership relational modeling for complex networks. In *MLSP*. IEEE.
- Nickel, M., Tresp, V., and Kriegel, H. (2011). A three-way model for collective learning on multi-relational data. In *ICML*.
- Nickel, M., Tresp, V., and Kriegel, H.-P. (2012). Factorizing YAGO: scalable machine learning for linked data. In *WWW*.
- Papalexakis, E., Faloutsos, C., and Sidiropoulos, N. (2012). Parcube: Sparse parallelizable tensor decompositions. In *Machine Learning and Knowledge Discovery in Databases*, pages 521–536. Springer.
- Papalexakis, E. E., Mitchell, T. M., Sidiropoulos, N. D., Faloutsos, C., Talukdar, P. P., and Murphy, B. (2013). Scoup-smt: Scalable coupled sparse matrix-tensor factorization. *arXiv preprint arXiv:1302.7043*.
- Piegorsch, W. W. (1992). Complementary log regression for generalized linear models. *The American Statistician*.
- Rai, P., Wang, Y., and Carin, L. (2015). Leveraging features and networks for probabilistic tensor decomposition. In *AAAI*.
- Rai, P., Wang, Y., Guo, S., Chen, G., Dunson, D., and Carin, L. (2014). Scalable Bayesian low-rank decomposition of incomplete multiway tensors. In *ICML*.
- Xu, Z., Yan, F., and Qi, Y. (2013). Bayesian nonparametric models for multiway data analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- Zhe, S., Qi, Y., Park, Y., Molloy, I., and Chari, S. (2013). Dintucker: Scaling up gaussian process models on multidimensional arrays with billions of elements. *arXiv preprint arXiv:1311.2663*.
- Zhe, S., Xu, Z., Chu, X., Qi, Y., and Park, Y. (2015). Scalable nonparametric multiway data analysis. In *AISTATS*.
- Zhou, M. (2015). Infinite edge partition models for overlapping community detection and link prediction. In *AISTATS*.
- Zhou, M., Hannah, L. A., Dunson, D., and Carin, L. (2012). Beta-negative binomial process and poisson factor analysis. In *AISTATS*.

---

# Computing Optimal Bayesian Decisions for Rank Aggregation via MCMC Sampling

---

David Hughes and Kevin Hwang and Lirong Xia  
Rensselaer Polytechnic Institute, Troy, NY, USA  
{hughed2,hwangk2}@rpi.edu, xial@cs.rpi.edu

## Abstract

We propose two efficient and general MCMC algorithms to compute optimal Bayesian decisions for Mallows’ model and Condorcet’s model w.r.t. any loss function and prior. We show that the mixing time of our Markov chain for Mallows’ model is polynomial in  $\varphi^{-k_{max}}$ ,  $d_{max}$ , and the input size, where  $\varphi$  is the dispersion of the model,  $k_{max}$  measures agents’ largest total bias in bipartitions of alternatives, and  $d_{max}$  is the maximum ratio between prior probabilities. We also show that in some cases the mixing time is at least  $\Theta(\varphi^{-k_{max}/2})$ . For Condorcet’s model, our Markov chain is rapid mixing for moderate prior distributions. Efficiency of our algorithms are illustrated by experiments on real-world datasets.

## 1 INTRODUCTION

In many social choice (a.k.a. rank aggregation) problems we want to compute an objectively optimal joint decision based on agents’ preferences. For example, the *Condorcet Jury Theorem* (Condorcet, 1785) studies how to select a “correct” leader in political elections when the votes are noisy perceptions of the ground truth. Principles and algorithms for social choice have also been used to aggregate rankings in meta-search engines (Dwork et al., 2001), recommender systems (Ghosh et al., 1999), crowdsourcing (Mao et al., 2013), semantic webs (Porello and Endriss, 2013), and peer grading for MOOC (Raman and Joachims, 2014).

In these social choice applications it is natural to take a Bayesian approach. Given a statistical model that describes agents’ noisy perception of the ground truth, a prior distribution, and a loss function that evaluates the joint decision w.r.t. the ground truth, we compute an optimal joint decision that has the minimum Bayesian expected loss w.r.t. the posterior distribution. This was recently formalized as a

*statistical decision-theoretic framework for social choice* by Azari Soufiani et al. (2014).

A major challenge in previous research, especially in the Bayesian approaches, is the high computational complexity of decision making. For example, the maximum likelihood estimator (MLE) of a popular ranking model called *Mallows’ model* (Mallows, 1957) is NP-hard to compute (Bartholdi et al., 1989). Computing optimal Bayesian decisions for rank aggregation is a hard combinatorial optimization problem because the parameter space is often discrete and its size is often exponential. Most previous work focused on designing efficient case-by-case algorithms for computing MLEs and MAPs of popular ranking models. However, the following question is left unanswered:

*Are there general and efficient algorithms that compute optimal Bayesian decisions for a wide range of rank aggregation problems?*

**Our contributions.** We give positive answers to this question for two popular ranking models: *Mallows’ model* (Mallows, 1957) and *Condorcet’s model* (Condorcet, 1785; Young, 1988) by proposing two general Markov chain Monte Carlo (MCMC) algorithms. Our algorithms work for any prior distribution, any decision space, and any loss function. In both algorithms, we first generate multiple samples by a Markov chain whose stationary distribution is the posterior distribution, which are used to compute the optimal decision that minimizes the empirical loss. Our Markov chains for both models are Metropolis-Hastings samplers (Metropolis et al., 1953; Hastings, 1970). For Mallows’ model, we apply a random transposition on adjacent pairs of alternatives in each step, and for Condorcet’s model, we adopt an *independent sampler*, which samples a candidate parameter with probability that is proportional to its likelihood in each step.

We prove that our Markov chains have good theoretical guarantees on the *mixing time*, which measures the rate of convergence to the stationary distribution and is closely related to the overall running time of the algorithms (Theorem 1). For Mallows’ model, we show that the mixing

time of our Markov chain is polynomial in  $\varphi^{-k_{max}}$ ,  $d_{max}$ , and the input size, where  $\varphi$  is the dispersion of the model,  $k_{max}$  measures agents’ largest total bias in bipartitions of alternatives (Theorem 3), and  $d_{max}$  is the maximum ratio between prior probabilities. We also show that in some cases the mixing time is at least  $\Theta(\varphi^{-k_{max}/2})$  (Proposition 1). For Condorcet’s model, our Markov chain is rapid mixing for moderate prior distributions—its mixing time is polynomial in the input size and  $(\ln \frac{d_{max}}{d_{max}-1})^{-1}$  (Theorem 5). Therefore, we can efficiently generate samples to estimate the Bayesian expected loss with high accuracy, for Mallows’ model when  $\varphi^{-k_{max}}$  and  $d_{max}$  are not too large, and for Condorcet’s model when  $d_{max}$  is not too large, for a wide range of social choice problems including those that are provably NP-hard to compute or even approximate (Theorem 2 and 4).

Computational and statistical efficiency of our algorithms are shown in preliminary experiments using real-world election data from Preflib (Mattei and Walsh, 2013) ([www.Preflib.org](http://www.Preflib.org)). The key observation is that for Mallows’ model, when the number of alternatives is at least 11 the brute-force search takes much more time than generating 10 million samples, based on which we can achieve a high precision for estimating the Bayesian loss and making the optimal Bayesian decision.

**Related Work and Discussions.** MCMC methods have been widely applied in Bayesian statistics (Smith and Roberts, 1993). However, obtaining non-trivial bounds on the mixing time is a “considerable challenge” (Jerrum and Sinclair, 1996). Our proofs involve novel applications of a wide range of analytical tools, which we believe to have independent interest. To the best of our knowledge, this is the first time that MCMC methods for Bayesian inference for ranking models have been analytically studied. The theoretical bounds on the mixing time of the proposed Markov chains are our main theoretical contribution.

Practically, the main advantage of our algorithms is their generality because they work for any decision space, any loss function, and any prior distribution. For many natural loss functions (e.g. those in Definition 4), no efficient algorithm was previously known. There have been much work on computing MLE and MAP for Mallows’ model and Condorcet’s model in (computational) social choice (Bartholdi et al., 1989; Hemaspaandra et al., 2005; Betzler et al., 2008), theory (Ailon et al., 2005; Kenyon-Mathieu and Schudy, 2007), and machine learning (Kuo et al., 2009; Long et al., 2010; Lu and Boutilier, 2011; Liu, 2011; Negahban et al., 2012; Azari Soufiani et al., 2012, 2013a,b; Raman and Joachims, 2015). Also see the experimental study by Ali and Meila (2012) for a comparison of 104 algorithms and combinations. Our algorithms are more general, as MLEs and MAPs are special combinations of decision space and loss function.

Specifically, we have not seen much work on MCMC algorithms for rank aggregation. The work that is closest to ours is by Raman and Joachims (2015), who proposed a different Markov chain for Mallows’ model and tested it on MOOC grading data. However, their paper did not analyze the mixing time. Diaconis and Hanlon (1992) proposed a Metropolis-Hastings algorithm to generate data according to a Mallows-like model that is based on the *Cayley distance*, which is different from both models studied in this paper. Moreover, it is not clear whether the algorithm by Diaconis and Hanlon can be leveraged to generate samples from the posterior distribution.

## 2 PRELIMINARIES

Let  $\mathcal{C}$  denote a set of  $m$  alternatives. Let  $\mathcal{L}(\mathcal{C})$  denote the set of *linear orders* over  $\mathcal{C}$ , that is, the set of all transitive, antisymmetric, and total binary relations. Let  $\mathcal{B}(\mathcal{C})$  denote the set of all possibly cyclic orders over  $\mathcal{C}$ , that is, all irreflexive, antisymmetric, and total binary relations over  $\mathcal{C}$ . Clearly  $\mathcal{L}(\mathcal{C}) \subseteq \mathcal{B}(\mathcal{C})$ . Each agent uses a (possibly cyclic) order over  $\mathcal{C}$  to represent her preferences. Let  $R$  denote the (*preference*) *profile* containing preferences from  $n$  agents. Given  $R$ , we want to make a joint decision from a decision space  $\mathcal{D}$ , which can be different from  $\mathcal{C}$ .

For any profile  $R$ , its *weighted majority graph*, denoted by  $\text{WMG}(R)$ , is a weighted directed graph whose vertices are  $\mathcal{C}$ , and there is an edge between each pair of alternatives  $(a, b)$  with weight  $w_R(a, b) = \#\{V \in R : a \succ_V b\} - \#\{V \in R : b \succ_V a\}$ . Clearly  $w_R(a, b) + w_R(b, a) = 0$ .

For any  $V, W \in \mathcal{B}(\mathcal{C})$ , we let  $\text{KT}(V, W)$  denote the *Kendall-tau distance* between  $V$  and  $W$ , that is, the number of different pairwise comparisons in  $V$  and  $W$ . In this paper, we focus on the following two ranking models.

**Definition 1** (Mallows’ model with fixed dispersion). *The parameter space is  $\Theta_M = \mathcal{L}(\mathcal{C})$ , the sample space  $\mathcal{S}_M$  is composed of  $n$  i.i.d. generated data in  $\mathcal{L}(\mathcal{C})$ , and for any  $W \in \mathcal{L}(\mathcal{C})$  and any profile  $R$ , we have  $\text{Pr}_M(R|W) = \prod_{V \in R} \left( \frac{1}{Z_M} \varphi^{\text{KT}(V, W)} \right)$ , where  $Z_M$  is the normalization factor such that  $Z_M = \sum_{U \in \mathcal{L}(\mathcal{C})} \varphi^{\text{KT}(U, W)}$ .*

**Definition 2** (Condorcet’s model with fixed dispersion). *The parameter space is  $\Theta_C = \mathcal{B}(\mathcal{C})$ , the sample space  $\mathcal{S}_C$  is composed of  $n$  i.i.d. generated data in  $\mathcal{B}(\mathcal{C})$ , and for any  $W \in \mathcal{B}(\mathcal{C})$  and any profile  $R$ , we have  $\text{Pr}_C(R|W) = \prod_{V \in R} \left( \frac{1}{Z_C} \varphi^{\text{KT}(V, W)} \right)$ , where  $Z_C$  is the normalization factor such that  $Z_C = \sum_{U \in \mathcal{B}(\mathcal{C})} \varphi^{\text{KT}(U, W)}$ .<sup>1</sup>*

In the above two definitions, the normalization factors  $Z_M$  and  $Z_C$  are independent of the selection of  $W$ . We now recall the *statistical decision-theoretic framework for social*

<sup>1</sup>Our results also work for the variant where the sample space is  $(\mathcal{L}(\mathcal{C}))^n$ .

choice defined by Azari Soufiani et al. (2014) to formulate the computational problem.

**Definition 3** ((Azari Soufiani et al., 2014)). A statistical decision-theoretic framework for social choice (SDT framework for short) is a tuple  $\mathcal{F} = (\mathcal{M}_C, \mathcal{D}, L)$ , where  $\mathcal{C}$  is the set of alternatives,  $\mathcal{M}_C = (\Theta, \text{Pr}, \mathcal{S})$  is a ranking model,  $\mathcal{D}$  is the decision space, and  $L : \Theta \times \mathcal{D} \rightarrow \mathbb{R}_{\geq 0}$  is a loss function that is easy to compute.

In this paper, we focus on computing the *Bayesian estimators*  $f_B$  of a SDT framework that minimizes the expected Bayesian loss for any prior and profile. More precisely, given a SDT framework  $\mathcal{F}$ , a prior over  $\Theta$ , and a profile  $R$ ,  $f_B(R) \in \min_{d \in \mathcal{D}} E_{\theta \sim \text{Pr}(\cdot | R)} L(\theta, d)$ . For example, the *maximum a posteriori (MAP)* is the Bayesian estimator for the SDT framework with  $\mathcal{D} = \Theta$  and the 0-1 loss function.

Natural choices of the decision space for ranking models are: (1)  $\mathcal{D} = \mathcal{C}$ , and  $f_B$  is called a *resolute voting rule*. (2)  $\mathcal{D} = 2^{\mathcal{C}} - \{\emptyset\}$ , and  $f_B$  is called a *irresolute voting rule*. (3)  $\mathcal{D} = \mathcal{L}(\mathcal{C})$ , and  $f_B$  is called a *preference function* or *social welfare function*.

In this paper, we focus on case (1)  $\mathcal{D} = \mathcal{C}$  and the exact Top- $k$  loss functions defined below. The proposed algorithms and theorems on the mixing time work for any SDT framework.

**Definition 4.** Let  $\mathcal{D} = \mathcal{C}$ . For any  $k \leq m - 1$ , the exact Top- $k$  loss function  $L_{E\text{Top-}k}$  is defined as: for any  $W \in \mathcal{B}(\mathcal{C})$  and  $d \in \mathcal{C}$ ,  $L_{E\text{Top-}k}(W, d) = 0$  if there exists  $A \subseteq \mathcal{C}$  such that  $|A| = k$ ,  $d \in A$ , and for all  $a \in A, b \in \mathcal{C} - A$  we have  $a \succ_W b$ ; otherwise  $L_{E\text{Top-}k}(W, d) = 1$ .

In words, the loss of an alternative  $d$  under the exact Top- $k$  loss function is 0 if  $d$  is clearly ranked within top  $k$  positions in the ground truth; otherwise the loss is 1.<sup>2</sup>

A Markov chain over a state space  $\Theta$  is characterized by a transition matrix  $P$  such that for any  $V, W \in \Theta$ ,  $P(V, W)$  is the probability for the next state to be  $W$  given that the current state is  $V$ . Therefore, for any  $V \in \Theta$ , we have  $\sum_{W \in \Theta} P(V, W) = 1$ . In this paper, the state space is the parameter space because we want to sample from the posterior distribution. The *stationary distribution*  $\pi$  of a Markov chain with transition matrix  $P$  is a probability distribution over  $\Theta$  such that for any  $V \in \Theta$  we have  $\sum_{W \in \Theta} \pi(W)P(W, V) = \pi(V)$ , that is,  $\pi$  (as a row vector) is a left eigenvector of  $P$  with eigenvalue 1.

Given a Markov chain with transition matrix  $P(V, W)$  and a unique stationary distribution  $\pi(\cdot)$ , the *variation distance* at time  $t$  w.r.t. starting state  $V$  is defined to be

$$\Delta_V(t) = \max_{S \subseteq \Theta} |P^t(V, S) - \pi(S)|$$

<sup>2</sup>We note that for some  $W \in \mathcal{B}(\mathcal{C})$  no alternative is clearly ranked within top  $k$ . For any  $W \in \mathcal{L}(\mathcal{C})$ , there are always  $k$  alternatives clearly ranked in top  $k$ .

where  $P^t(V, S)$  is the probability for the Markov chain starting at  $V$  to end in a state in  $S$  after  $t$  steps.

The convergence rate of a Markov chain to the stationary distribution is measured by its *mixing time*  $\tau_V(\epsilon)$ , which is the number of steps that guarantee a variation distance below  $\epsilon$ . Formally, we define

$$\tau_V(\epsilon) = \min\{t : \Delta_V(t') \leq \epsilon \text{ for all } t' \geq t\}$$

Let  $\tau(\epsilon)$  denote the maximum mixing time for all starting states, that is,  $\tau(\epsilon) = \max_V \tau_V(\epsilon)$ .

Our algorithms first use a Markov chain sampler  $\mathfrak{M}$  that runs a Markov chain for multiple steps to generate samples from the parameter space  $\Theta$ , then compute the optimal decision w.r.t. these samples. Formally, we use Algorithm 1 to estimate the expected Bayesian loss of all decisions, then choose one with minimum expected loss.

---

**Algorithm 1** CompBayesianLoss

---

- 1: **Input:** a profile  $R$ , a SDT framework  $\mathcal{F} = (\mathcal{M}_C, \mathcal{D}, L)$  with prior  $\text{Pr}(\cdot)$ , a Markov chain sampler  $\mathfrak{M}$  over  $\Theta$  whose stationary distribution is  $\text{Pr}(\cdot | R)$ , and a decision  $d \in \mathcal{D}$ .
  - 2: Use  $\mathfrak{M}$  to generate  $N$  independent samples, denoted by  $Q$ .
  - 3: **return**  $\sum_{W \in Q} L(W, d) / |Q|$ .
- 

It is well-known that the mixing time is closely related to the running time of approximate algorithms based on samples generated from the Markov chain. For SDT frameworks, this relation is formalized in Theorem 1. For completeness we include a short proof.

**Theorem 1.** For any  $d \in \mathcal{D}$  and any  $\epsilon > 0, \delta > 0$ , Algorithm 1 can compute the expected Bayesian loss of  $d$  with no more than  $\epsilon$  additive error with probability at least  $1 - \delta$  in  $O(\frac{l_{max}^2}{\epsilon^2} \ln \delta^{-1} \tau(\frac{\epsilon}{2l_{max}}) \eta)$  time, where  $l_{max}$  is the maximum loss in  $L$ ,  $\tau(\cdot)$  is the mixing time of the Markov chain used by  $\mathfrak{M}$ , and  $\eta$  is running time for one step in  $\mathfrak{M}$ .

**Proof:** Let  $\pi$  denote the posterior distribution over  $\Theta$  given  $R$  and let  $\pi^*$  denote the distribution by the Markov chain sampler. We first prove that using  $O(\frac{l_{max}^2}{\epsilon^2} \ln \delta^{-1})$  independent samples, the output of Algorithm 1 is no more than  $\epsilon/2$  away from  $E_{V \sim \pi^*} L(V, d)$ . We then choose a sampler  $\mathfrak{M}$  with mixing time  $\frac{\epsilon}{2l_{max}}$  and show that  $|E_{V \sim \pi^*} L(V, d) - E_{V \sim \pi} L(V, d)| \leq \frac{\epsilon}{2}$ .

Let  $X^1, \dots, X^N$  denote  $N$  i.i.d. random variables distributed as  $L(V, d)$ , where  $V$  is generated from  $\pi^*$ . Let  $Y^N = (\sum_{i=1}^N X^i) / N$ . Because for any  $\theta \in \Theta$  and  $d \in \mathcal{D}$ ,  $0 \leq L(\theta, d) \leq l_{max}$ , we have  $\text{Var}(X^1) \leq l_{max}^2$  and  $\text{Var}(Y^N) \leq l_{max}^2 / N$ . Also it is easy to check that  $E_{V \sim \pi^*} L(V, d) = E(X^1) = E(Y^N)$ . Therefore, by Chebyshev's inequality we have:  $\Pr(|Y^N - E(Y^N)| \geq$

$\frac{\epsilon}{2}) \leq \frac{4\text{Var}(Y^N)}{\epsilon^2} \leq \frac{4l_{max}^2}{\epsilon^2 N}$ . When  $N \geq \frac{16}{3} \frac{l_{max}^2}{\epsilon^2}$  we have  $\Pr(|Y^N - E(Y^N)| \geq \frac{\epsilon}{2}) \leq \frac{3}{4}$ . This can be leveraged to an algorithm that outputs an estimation to  $E(X^1)$  with no more than  $\frac{\epsilon}{2}$  additive error with probability at least  $1 - \delta$ , using  $O(\frac{l_{max}^2}{\epsilon^2} \ln \delta^{-1})$  calls to the sampler  $\mathfrak{M}$ . For any  $\mathfrak{M}$  with mixing time  $\tau(\frac{\epsilon}{2l_{max}})$ , we have  $|E_{V \sim \pi^*} L(V, d) - E_{V \sim \pi} L(V, d)| \leq \sum_{V \in \Theta} L(V, d) |\pi(V) - \pi^*(V)| \leq l_{max} \frac{\epsilon}{2l_{max}} = \frac{\epsilon}{2}$ .

Therefore, the total running time of Algorithm 1 is  $O(\frac{l_{max}^2}{\epsilon^2} \ln \delta^{-1} \tau(\frac{\epsilon}{2l_{max}}) \eta)$ .  $\square$

In the remainder of this paper we focus on the Markov chains for Mallows' model and Condorcet's model. For both models, we will design *Metropolis-Hastings sampling algorithms* (Metropolis et al., 1953; Hastings, 1970), which work as follows. For each state  $V$  we first generate a candidate  $W$  for the next state from a proposal distribution  $p_V(\cdot)$ . Then, with probability  $\min\{1, \frac{\pi(W)p_W(V)}{\pi(V)p_V(W)}\}$  the next state is  $W$ ; otherwise the next state remains at  $V$ .

### 3 MARKOV CHAIN FOR MALLOWS' MODEL

To motivate the study, we first prove that the expected Bayesian loss is hard to approximate for Mallows' model w.r.t. the exact Top-1 loss function and uniform prior. In the BAYESIANLOSS problem, we are given a SDT framework, a prior, and a decision  $d \in \mathcal{D}$ . We are asked to compute the expected Bayesian loss of  $d$ .<sup>3</sup>

**Theorem 2.** *If BAYESIANLOSS for Mallows' model w.r.t. the exact Top-1 loss function and the uniform prior has a polynomial-time approximation algorithm with constant approximation ratio, then  $\mathbf{P} = \mathbf{NP} = \mathbf{P}_{||}^{\mathbf{NP}}$ .*

$\mathbf{P}_{||}^{\mathbf{NP}}$  is the class of problems that can be computed by a  $\mathbf{P}$  oracle machine using polynomial number of parallel access to  $\mathbf{NP}$  oracles.  $\mathbf{P}_{||}^{\mathbf{NP}}$  contains  $\mathbf{NP}$  and  $\mathbf{co-NP}$ .

**Proof:** The hardness is proved by a reduction from the KEMENYWINNER problem, which is  $\mathbf{NP}$ -hard (Bartholdi et al., 1989) and  $\mathbf{P}_{||}^{\mathbf{NP}}$ -complete (Hemaspaandra et al., 2005). Given a profile  $R$ , for any alternative  $c$ , the *Kemeny score* of  $c$  is the smallest Kendall-tau distance between the profile and any linear order where  $c$  is ranked at the top. An alternative with the minimum Kemeny score is called a *Kemeny winner*. In a KEMENYWINNER problem, we are given a profile  $R$  and an alternative  $c$ , and we are asked if  $c$  is a Kemeny winner.

Given a KEMENYWINNER instance  $(R, c)$ , we construct a BAYESIANLOSS instance where there is a new alternative  $d$  and the profile  $R'$  satisfies that  $\text{WMG}(R')$  equals to  $\text{WMG}(R)$  plus the edges  $d \rightarrow a$  for all  $a \neq c$ , whose

<sup>3</sup>This problem is known to be  $\mathbf{NP}$ -hard and  $\mathbf{P}_{||}^{\mathbf{NP}}$ -hard to compute exactly (Procaccia et al., 2012; Azari Soufiani et al., 2014).

weights are 1 (if weights on edges in  $\text{WMG}(R)$  are odd) or 2 (if weights on edges in  $\text{WMG}(R)$  are even). Given any constant  $\alpha > 1$ , we let  $\varphi = \frac{\alpha^2}{2(m!)^2}$ . We note that  $\varphi$  can be represented using polynomial number of bits. This instance can be constructed in polynomial time using McGarvey's trick (McGarvey, 1953).

Clearly  $d$  is a Kemeny winner in  $R'$ . If the KEMENYWINNER is a "yes" instance, then  $c$  is also a Kemeny winner in  $R'$  (where  $d$  is ranked in the second place in the winning ranking). The Bayesian expected loss of  $d$  is at least  $\frac{1}{m!}$  because in at least one ranking  $d$  is not at the top.

If the KEMENYWINNER is a "no" instance, then  $d$  is the unique Kemeny winner. Let  $V$  denote a ranking where  $d$  is ranked in the top and  $\text{KT}(V, R')$  equals to the Kemeny score of  $d$ . It is easy to check that for any ranking  $W$  where  $d$  is not ranked in the top,  $\text{KT}(V, R') \leq \text{KT}(W, R') - 1$ , which means that  $\frac{\Pr(W|R')}{\Pr(V|R')} \leq \varphi$ . Therefore, the posterior probability that  $d$  is not ranked in the top, which equals to the expected Bayesian loss of  $d$ , is at most  $\frac{(m!-1)\varphi}{1+(m!-1)\varphi}$ .

When  $\varphi = \frac{\alpha^2}{2(m!)^2}$ , we have  $\frac{(m!-1)\varphi}{1+(m!-1)\varphi} < \frac{\alpha^2}{m!}$ .

It follows that if there exists a polynomial-time  $\alpha$ -approximation algorithm for BAYESIANLOSS, then we can use this algorithm to solve KEMENYWINNER in polynomial time: if the output of the algorithm is no more than  $\frac{\alpha}{m!}$  then the KEMENYWINNER instance is a "no" instance; otherwise the KEMENYWINNER instance is a "yes" instance. This means that  $\mathbf{P} = \mathbf{NP} = \mathbf{P}_{||}^{\mathbf{NP}}$ .  $\square$

We now present the Markov chain  $\mathfrak{M}_M$  for Mallows' model in Algorithm 2, which runs  $\mathfrak{M}_M$  for  $N$  steps. In each step, we first apply a random transposition of adjacent alternatives in the current state  $V$  (changing  $d \succ c$  to  $c \succ d$ ) to obtain a candidate ranking  $W$ , then with probability  $\frac{1}{2} \min\{\frac{\Pr_M(W)}{\Pr_M(V)} \varphi^{R[d \succ c] - R[c \succ d]}, 1\}$  we let  $V = W$ , otherwise the next state stays at  $V$ , where  $R[c \succ d]$  is the number of times  $c \succ d$  in  $R$ . The  $\frac{1}{2}$  factor is a popular trick to prove bounds on the mixing time in Lemma 2.

---

#### Algorithm 2 Markov chain $\mathfrak{M}_M$ for Mallows' model.

---

- 1: **Inputs:** a profile  $R$ , a prior  $\Pr_M$ , an initial ranking  $V$ , and the number of iterations  $N$ .
  - 2: **for**  $t=1$  **to**  $N$  **do**
  - 3:   Switch a pair of adjacent alternatives uniformly at random (from  $d \succ c$  to  $c \succ d$ ). Let  $W$  denote the new ranking.
  - 4:   With probability  $\frac{1}{2} \min\{\frac{\Pr_M(W)}{\Pr_M(V)} \varphi^{R[d \succ c] - R[c \succ d]}, 1\}$  let  $V = W$ .
  - 5: **end for**
  - 6: **return**  $V$ .
- 

For any profile  $R$ , let  $k_{max}$  denote the max cut of the undirected  $\text{WMG}(R)$ . That is,  $k_{max} = \max_{A \subseteq C} \sum_{a \in A, b \in C-A} |w_R(a, b)|$ . Let  $d_{max}$  denote the

maximum ratio between prior probabilities. That is,  $d_{max} = \max_{V, W \in \Theta} \frac{\text{Pr}_M(V)}{\text{Pr}_M(W)}$ .

**Theorem 3.** *The mixing time of the Markov chain in Algorithm 2 for any starting state is  $O(m^4 d_{max}^3 \varphi^{-k_{max}} (nm^3 \log m \ln \varphi^{-1} + \ln \epsilon^{-1}))$ .*

**Proof:** It is easy to check that the Markov chain  $\mathfrak{M}_M$  in Algorithm 2 is finite, ergodic, reversible and the transition matrix  $P_M(V, W)$  is diagonally dominant. Therefore, all eigenvalues of  $P_M(V, W)$  are real and positive, and the largest one is 1. The following lemma shows that the mixing time is closely related to the *spectral gap*  $1 - \lambda_{max}$ , where  $\lambda_{max}$  is the second largest eigenvalue of  $P_M(V, W)$ . It is easy to verify that the stationary distribution  $\pi_M$  equals to the posterior distribution  $\text{Pr}_M(\cdot | R)$ .

**Lemma 1** (e.g. (Sinclair, 1992)).

$$(i) \tau_V(\epsilon) \leq (1 - \lambda_{max})^{-1} (\ln \pi(V)^{-1} + \ln \epsilon^{-1});$$

$$(ii) \max_{V \in \Theta} \tau_V(\epsilon) \geq \frac{1}{2} (1 - \lambda_{max})^{-1} \ln(2\epsilon)^{-1}.$$

It is often hard to directly obtain lower bounds on the spectral gap. We will take the *canonical path* approach, whose idea is the following. Any reversible Markov chain can be visualized as an undirected graph  $G$  where the vertices are  $\Theta$  and the weight on the edge between  $V$  and  $W$  is  $Q(V, W) = \pi(V)P(V, W)$ . For each pair of states  $V, W \in \Theta$ , we fix a directed path (canonical path) from  $V$  to  $W$  in  $G$ , denoted by  $\gamma_{VW}$ . Let  $|\gamma_{VW}|$  denote the length of  $\gamma_{VW}$ . Let  $\Gamma$  denote the set of all canonical paths defined above, one for each pair  $(V, W)$ . The maximum *loading* of a single edge in  $\Gamma$  provides a lower bound on the spectral gap, thus it can be used to upper-bound the mixing time. Formally, this was proved by Sinclair (1992) in the following Lemma.

**Lemma 2** ((Sinclair, 1992; Jerrum and Sinclair, 1996)). *Let  $\mathfrak{M}$  be a finite, reversible, and ergodic Markov chain with loop probabilities  $P(V, V) \geq \frac{1}{2}$  for all states  $V$ . Let  $\Gamma$  be a set of canonical paths with maximum edge loading*

$$\rho = \max_e \frac{1}{Q(e)} \sum_{\gamma_{VW} \ni e} \pi(V)\pi(W)|\gamma_{VW}|$$

*Then the mixing time satisfies  $\tau_V(\epsilon) \leq \rho(\ln \pi(V)^{-1} + \ln \epsilon^{-1})$  for all  $V \in \Theta$ .*

To apply Lemma 2, we consider the following canonical paths  $\Gamma_M$  for  $\mathfrak{M}_M$ . We note that the graph  $G$  for canonical paths, whose vertices are rankings over alternatives, is different from the weighted majority graph, whose vertices are the alternatives.

**Definition 5.** *In the canonical paths  $\Gamma_M$ , for any pair of different rankings  $V, W \in \mathcal{L}(\mathcal{C})$ ,  $\gamma_{VW}$  contains the rankings obtained in  $m - 1$  stages of adjacent transpositions, where in stage  $k$ , the alternative ranked at the  $k$ -th position in  $W$  is moved up to the  $k$ -th position in  $V$ .*

W.l.o.g. let  $W = [a_1 \succ \dots \succ a_m]$ . In stage 1, we apply the minimum number of adjacent transpositions on  $V$  to move  $a_1$  to the top position. This process passes through no more than  $m - 1$  rankings, and let  $V_1$  denote the ranking at the end of the process, where  $a_1$  is ranked at the top position and the other part of  $V_1$  is the same as in  $V$ . In stage 2, if  $a_2$  is not already ranked at the second position of  $V_1$ , then we apply the minimum number of adjacent transpositions on  $V_1$  to move  $a_2$  to the second position. The process continues until we reach  $W$ . For example, when  $m = 4$ ,  $V = [a_4 \succ a_1 \succ a_3 \succ a_2]$ , and  $W = [a_1 \succ a_2 \succ a_3 \succ a_4]$ , we have  $\gamma_{VW} = V \rightarrow [a_1 \succ a_4 \succ a_3 \succ a_2] \rightarrow [a_1 \succ a_4 \succ a_2 \succ a_3] \rightarrow [a_1 \succ a_2 \succ a_4 \succ a_3] \rightarrow W$ .

It is not hard to see that  $|\gamma_{VW}| \leq m^2$ . For any edge  $e = E \rightarrow E'$  in a canonical path, we have  $Q_M(E, E') = \pi_M(E)P_M(E, E') = \pi_M(E')P_M(E', E) \geq \frac{1}{2(m-1)} \min\{\pi_M(E), \pi_M(E')\}$ . Therefore, we have

$$\rho \leq \max_{e = E \rightarrow E'} \left( \frac{2m^2(m-1)}{\min\{\pi_M(E), \pi_M(E')\}} \sum_{\gamma_{VW} \ni e} \pi_M(V)\pi_M(W) \right) \quad (1)$$

**Lemma 3.** *Given the canonical paths  $\Gamma_M$  defined in Definition 5 and any edge  $e = E \rightarrow E'$ , we have:*

- (i)  $\sum_{V, W: \gamma_{VW} \ni e} \pi_M(V)\pi_M(W)/\pi_M(E) \leq m\varphi^{-k_{max}}$ , and
- (ii)  $\sum_{V, W: \gamma_{VW} \ni e} \pi_M(V)\pi_M(W)/\pi_M(E') \leq m\varphi^{-k_{max}}$ .

**Proof:** Let  $E = [T \succ d \succ c \succ B]$  and  $E' = [T \succ c \succ d \succ B]$ . For any  $0 \leq k \leq |T|$ , we let  $T_k$  denote the top  $k$  ordering of  $T$  and let  $T_k^*$  denote the remaining ordering. That is, for any  $k \leq |T|$  we have  $T = [T_k \succ T_k^*]$ . It is easy to check that  $e \in \gamma_{VW}$  if and only if there exists  $0 \leq k \leq |T| \leq m - 2$  such that the following conditions hold.

- (1)  $T_k^* \succ d \succ B$  and  $d \succ c$  hold in  $V$ . Let  $\mathcal{V}_k$  denote the set of all such  $V$ 's.
- (2) The top  $k + 1$  alternatives in  $W$  are ranked as  $[T_k \succ c]$ . Let  $\mathcal{W}_k$  denote the set of all such  $W$ 's.

We first prove the inequality for  $E$ . Let  $A_k$  denote the alternatives in  $T_k$ , let  $A_k^*$  denote the alternatives in  $T_k^*$  plus  $d$ , let  $S$  denote the alternatives in  $B$ , and let  $\bar{A}_k = \mathcal{C} - A_k$ . For each pairwise comparison  $a \succ_E b$  in  $E$ , either we have (1)  $a \succ_V b$  for all  $V \in \mathcal{V}_K$  or (2)  $a \succ_W b$  for all  $W \in \mathcal{W}_K$ . This relationship is shown in Table 1.

For example, “ $W$ ” at  $(A_k, A_k)$  in Table 1 means that for all  $W \in \mathcal{W}_k$ ,  $(a, b) \in A_k \times A_k$ ,  $a \succ_W b$  if and only if  $a \succ_E b$ .  $(c, c)$  is marked N/A because the pairwise comparison between  $c$  and  $c$  is not well defined.

For any  $J \subseteq \mathcal{L}(\mathcal{C}) \times \mathcal{L}(\mathcal{C})$  and  $V, W \in \mathcal{L}(\mathcal{C})$ , we let  $D_J(V, W)$  denote the number of different pairwise comparisons between  $V$  and  $W$  for all pairs  $\{a, b\}$  such that

|         | $A_k$ | $A_k^*$ | $c$ | $S$ |
|---------|-------|---------|-----|-----|
| $A_k$   | $W$   | $W$     | $W$ | $W$ |
| $A_k^*$ | $W$   | $V$     | $V$ | $V$ |
| $c$     | $W$   | $V$     | N/A | $W$ |
| $S$     | $W$   | $V$     | $W$ | $V$ |

Table 1: Pairwise comparisons in  $E$  that are the same as in  $V \in \mathcal{V}_k$  or  $W \in \mathcal{W}_k$ .

$(a, b) \in J$  or  $(b, a) \in J$ . Formally,  $D_J(V, W) = \#\{\{a, b\} : [(a, b) \in J \text{ or } (b, a) \in J] \text{ and } [[a \succ_V b \text{ and } b \succ_W a] \text{ or } [a \succ_W b \text{ and } b \succ_V a]]\}$ . In other words,  $D_J(V, W)$  is the Kendall-tau distance between the restriction of  $V$  on  $J$  and the restriction of  $W$  on  $J$ . We note that any unordered pair of alternatives  $\{a, b\}$  is counted only once in  $J$ . In particular, for any  $a \in \mathcal{C}$  and  $A \subset \mathcal{C}$ ,  $D_{\{a\} \times \{a\}}(V, W) = 0$  and  $D_{(\{a\} \cup A) \times (\{a\} \cup A)}(V, W) = D_{A \times (\{a\} \cup A)}(V, W) = D_{(\{a\} \cup A) \times A}(V, W)$ . We have

$$\begin{aligned}
& \sum_{V, W: \gamma_V W \ni e} \pi_M(V) \pi_M(W) / \pi_M(E) \\
&= \sum_{0 \leq k \leq |T|} \sum_{V \in \mathcal{V}_k, W \in \mathcal{W}_k} \pi_M(V) \pi_M(W) / \pi_M(E) \\
&\leq \sum_{0 \leq k \leq |T|} \sum_{V \in \mathcal{V}_k, W \in \mathcal{W}_k} d_{max}^3 \frac{\Pr_M(R|V) \Pr_M(R|W)}{\Pr_M(R|E) \sum_{U \in \mathcal{L}(\mathcal{C})} \Pr_M(R|U)} \\
&= \sum_{0 \leq k \leq |T|} \sum_{V \in \mathcal{V}_k, W \in \mathcal{W}_k} \frac{d_{max}^3 f_1(V, E) f_2(W, E)}{\sum_{U \in \mathcal{L}(\mathcal{C})} \varphi^{KT(U, R)}} \quad (3)
\end{aligned}$$

where  $f_1(V, E) = \varphi^{D_{A_k \times \mathcal{C}}(V, R) + D_{\{c\} \times S}(V, R)}$  and  $f_2(W, E) = \varphi^{D_{A_k^* \times \bar{A}_k}(W, R) + D_{S \times S}(W, R)}$ . (2) is due to the following lemma.

**Lemma 4.** For any ranking model, any  $V \in \Theta$ , and any profile  $R$ , we have  $\frac{\Pr(V|R)}{d_{max}} \leq \frac{\Pr(R|V)}{\sum_{U \in \Theta} \Pr(R|U)} \leq d_{max} \Pr(V|R)$ .

For (3), according to Table 1, we have (for all grids with “V” in Table 1)

$$\begin{aligned}
& D_{A_k^* \times \bar{A}_k}(E, R) + D_{S \times S}(E, R) \\
&= D_{A_k^* \times \bar{A}_k}(V, R) + D_{S \times S}(V, R)
\end{aligned}$$

and (for all grids with “W” in Table 1)

$$\begin{aligned}
& D_{A_k \times \mathcal{C}}(E, R) + D_{\{c\} \times S}(E, R) \\
&= D_{A_k \times \mathcal{C}}(W, R) + D_{\{c\} \times S}(W, R)
\end{aligned}$$

We note that for any  $U \in \mathcal{L}(\mathcal{C})$ ,  $\Pr(R|U) \propto \varphi^{KT(U, R)}$  and

$$\begin{aligned}
KT(U, R) &= D_{A_k \times \mathcal{C}}(U, R) + D_{A_k^* \times \bar{A}_k}(U, R) \\
&\quad + D_{\{c\} \times S}(U, R) + D_{S \times S}(U, R)
\end{aligned}$$

Therefore,

$$\begin{aligned}
KT(E, R) &= D_{A_k^* \times \bar{A}_k}(V, R) + D_{S \times S}(V, R) \\
&\quad + D_{A_k \times \mathcal{C}}(W, R) + D_{\{c\} \times S}(W, R) \quad (4)
\end{aligned}$$

(3) follows after substituting (4) into (2).

We next prove that  $\sum_{V \in \mathcal{V}_k, W \in \mathcal{W}_k} \frac{f_1(V, E) f_2(W, E)}{\sum_{U \in \mathcal{L}(\mathcal{C})} \varphi^{KT(U, R)}} \leq \varphi^{-k_{max}}$ . To do so, we define a function  $g : \mathcal{V}_k \times \mathcal{W}_k \rightarrow \mathcal{L}(\mathcal{C})$  as follows: for any  $(V, W) \in \mathcal{V}_k \times \mathcal{W}_k$ ,  $g(V, W)$  is obtained from  $V$  by applying a permutation over  $A_k^* \cup S$  so that the preferences of  $g(V, W)$  over  $A_k^* \cup S$  become the preferences of  $W$  over  $A_k^* \cup S$ , while the other pairwise comparisons stay the same as in  $V$ . This means that the positions of  $A_k \cup \{c\}$  in  $g(V, W)$  are the same as in  $V$ . It is not hard to verify that for all pairs  $(V_1, W_1) \neq (V_2, W_2)$ , we have  $g(V_1, W_1) \neq g(V_2, W_2)$ , which means that  $\{g(V, W) : V \in \mathcal{V}_k, W \in \mathcal{W}_k\} \subseteq \mathcal{L}(\mathcal{C})$ .

**Claim 1.** For any  $(V, W) \in \mathcal{V}_k \times \mathcal{W}_k$ ,  $f_1(V, E) f_2(W, E) \leq \varphi^{KT(g(V, W), R) - k_{max}}$ .

**Proof:** By the definition of  $g(V, W)$  we have  $D_{A_k \times (A_k \cup \{c\})}(V, R) = D_{A_k \times (A_k \cup \{c\})}(g(V, W), R)$  and  $D_{A_k^* \times (A_k^* \cup S)}(W, R) = D_{A_k^* \times (A_k^* \cup S)}(g(V, W), R)$ . Therefore,

$$\begin{aligned}
& f_1(V, E) f_2(W, E) / \varphi^{g(V, W)} \\
&= \frac{\varphi^{D_{A_k \times (A_k^* \cup S)}(V, R) + D_{\{c\} \times S}(V, R) + D_{\{c\} \times A_k^*}(W, R)}}{\varphi^{D_{(A_k \cup \{c\}) \times (A_k^* \cup S)}(g(V, W), R)}} \\
&\leq \varphi^{-\sum_{a \in A_k \cup \{c\}, b \in A_k^* \cup S} |w_R(a, b)|} \leq \varphi^{-k_{max}}
\end{aligned}$$

□

By Claim 1 we have

$$\begin{aligned}
& \sum_{0 \leq k \leq |T|} \sum_{V \in \mathcal{V}_k, W \in \mathcal{W}_k} \frac{f_1(V, E) f_2(W, E)}{\sum_{U \in \mathcal{L}(\mathcal{C})} \varphi^{KT(U, R)}} \\
&\leq \sum_{0 \leq k \leq |T|} \frac{\sum_{V \in \mathcal{V}_k, W \in \mathcal{W}_k} f_1(V, E) f_2(W, E)}{\sum_{V \in \mathcal{V}_k, W \in \mathcal{W}_k} \varphi^{KT(g(V, W), R)}} \\
&\leq \sum_{0 \leq k \leq |T|} \max_{V \in \mathcal{V}_k, W \in \mathcal{W}_k} f_1(V, E) f_2(W, E) / \varphi^{KT(g(V, W), R)} \\
&\leq m \varphi^{-k_{max}}
\end{aligned}$$

This proves the inequality for  $E$ . The inequality for  $E'$  is proved similarly by letting  $A_k^*$  denote the alternatives in  $T_k^*$  and letting  $S$  denote the alternatives in  $B$  plus  $d$ . □

Combining Lemma 3 and inequality (1), we have  $\rho \leq 2m^4 d_{max}^3 \varphi^{-k_{max}}$ . We also note that for any state  $V$ ,  $\pi(V) \geq \varphi^{nm^2} / m!$ , which means that  $\ln \pi(V)^{-1}$  is  $O(nm^3 \log m \ln \varphi^{-1})$ . The theorem follows after applying Lemma 2. □



**Remarks:** The upper bound proved in Theorem 3 is polynomial in  $m, n, d_{max}, \ln \epsilon^{-1}$ , and is exponential in  $k_{max}$  (with base  $\varphi^{-1}$ ). Therefore, the algorithm is efficient if  $d_{max}$  and  $\varphi^{-k_{max}}$  are small, that is, either  $\varphi$  is close to 1 or  $k_{max}$  is small. The next proposition shows that the mixing time of  $\mathfrak{M}_M$  is sometimes  $\Omega(m\varphi^{-k_{max}/2})$ .

**Proposition 1.** *There exists a constant  $\alpha$  so that for any  $m \geq 3$ , there exists a profile  $R$  and the mixing time of  $\mathfrak{M}_M$  is at least  $\alpha m \varphi^{-k_{max}/2} \ln(2\epsilon)^{-1}$ .*

**Proof:** For any  $m \geq 3$  and any even number  $l$  we can construct a profile  $R^l$  with polynomial many votes using McGarvey’s trick (McGarvey, 1953) such that the  $\text{WMG}(R^l)$  contains only three edges:  $a_1 \rightarrow a_2, a_2 \rightarrow a_3, a_3 \rightarrow a_1$ , and the weight on all three edges is  $l$ .

It is easy to check that  $k_{max} = 2l$ . We prove the lower bound on the mixing time by applying Lemma 2(ii) and the conductance approach.

**Definition 6** (Sinclair and Jerrum (1989)). *The conductance of a Markov chain  $\mathfrak{M}$  is defined as*

$$\Phi(\mathfrak{M}) = \min_{S \subseteq \Theta: \pi(S) \leq 1/2} \frac{Q(S, \bar{S})}{\pi(S)},$$

where  $Q(S, \bar{S}) = \sum_{V \in S, W \in \bar{S}} Q(V, W)$ .

The spectral gap is related to the conductance in the following lemma proved by Sinclair and Jerrum (1989).

**Lemma 5** ((Sinclair and Jerrum, 1989)). *For any reversible Markov chain whose conductance is  $\Phi$ , the second eigenvalue  $\lambda_1$  satisfies  $1 - 2\Phi \leq \lambda_1 \leq 1 - \frac{\Phi^2}{2}$ .*

We recall that all eigenvalues of  $\mathfrak{M}_M$  are non-negative. Therefore, the spectral gap of  $\mathfrak{M}_M$  is  $1 - \lambda_1$ , which is at most  $2\Phi$ . The following claim gives an upper bound on the conductance for all  $R^l$ .

**Claim 2.** *There exists  $\beta > 0$  so that for all even number  $l$ ,  $\Phi(\mathfrak{M}_M) \leq \beta \frac{1}{m} \varphi^{k_{max}/2}$  for all  $R^l$ .*

**Proof:** We let  $S \subseteq \Theta$  denote the set of rankings where  $a_1 \succ a_2 \succ a_3$ . For any  $V \in S$  and  $W \in \bar{S}$ , if  $P(V, W) > 0$  then either  $a_2 \succ_W a_1 \succ_W a_3$  or  $a_1 \succ_W a_3 \succ_W a_2$ , which means that  $Q(V, \bar{S})/\pi(V) \leq \frac{1}{m-1} \varphi^{k_{max}/2}$ . Therefore,  $\frac{Q(S, \bar{S})}{\pi(S)} = \frac{\sum_{V \in S} Q(V, \bar{S})}{\sum_{V \in S} \pi(V)} \leq \frac{1}{m-1} \varphi^{k_{max}/2}$ . It is easy to check that  $1/6 \leq \pi(S) \leq 1/3$ , which means that there exists  $\beta > 0$  so that  $\Phi(\mathfrak{M}_M) \leq \frac{Q(S, \bar{S})}{\pi(S)} \leq \beta \frac{1}{m} \varphi^{k_{max}/2}$ .  $\square$

Combining Lemma 5 and Claim 2 we have  $1 - \lambda_{max} = 1 - \lambda_1 \leq \beta \frac{1}{m} \varphi^{k_{max}/2}$ . It follows from Lemma 1(ii) that  $\max_V \tau_V(\epsilon) \geq \frac{\beta}{2} m \varphi^{-k_{max}/2} \ln(2\epsilon)^{-1}$ .  $\square$

## 4 MARKOV CHAIN FOR CONDORCET’S MODEL

For Condorcet’s model it has been shown by Young (1988) that for any  $W \in \mathcal{B}(\mathcal{C})$  and any profile  $R$ ,  $\Pr_{\mathcal{C}}(R|W) \propto$

$\prod_{a \succ_W b} \left(\frac{\varphi}{1-\varphi}\right)^{R[a \succ b]}$ , where we recall that  $R[a \succ b]$  is the number of times  $a \succ b$  in  $R$ . This leads to the following observation.

**Proposition 2.** *Let  $R$  denote a profile of binary relations.  $V \in \mathcal{B}(\mathcal{C})$  maximizes the likelihood if and only if for any pair of alternatives  $(a, b)$ , we have  $(R[a \succ b] > R[b \succ a]) \Rightarrow (a \succ_V b)$ .*

An immediate corollary is that for any profile  $R$ , computing the MLE is in  $\mathbf{P}$ . While computing the expected Bayesian loss w.r.t. the exact Top-1 loss function is in  $\mathbf{P}$  (Young, 1988; Elkind and Shah, 2014; Azari Soufiani et al., 2014), for some natural loss functions computing the minimum expected Bayesian loss is  $\mathbf{NP}$ -hard. Formally, in a  $\text{MIN-BAYESIANLOSS}$  problem, we are given a SDT framework, a prior, a decision  $d \in \mathcal{D}$ , and a number  $l$ . We are asked whether there exists a decision whose expected Bayesian loss is no more than  $l$ .

**Theorem 4.**  *$\text{MINBAYESIANLOSS}$  can be computed in polynomial time for Condorcet’s model w.r.t.  $L_{\text{ETop-}k}$  and the uniform prior for any fixed  $k$ . It is  $\mathbf{NP}$ -hard to compute  $\text{MINBAYESIANLOSS}$  for Condorcet’s model w.r.t.  $L_{\text{ETop-}\frac{m}{2}}$  and the uniform prior for even  $m$ .*

**Proof:** For any fixed  $k$ , the Bayesian loss of any decision can be computed by enumerating all combinations of alternatives ranked at top  $k$  positions in the ground truth, the probability of which can be computed by Claim 3 below.

We prove the  $\mathbf{NP}$ -hardness of  $\text{MINBAYESIANLOSS}$  for Condorcet’s model w.r.t.  $L_{\text{ETop-}\frac{m}{2}}$  by a reduction from an  $\mathbf{NP}$ -hard problem called  $\text{ONEWAYBISECTION}$  (Feige and Yahalom, 2003). In a  $\text{ONEWAYBISECTION}$  instance, we are given an oriented graph  $G = (\mathcal{V}, \mathcal{E})$  with  $m$  vertices, where  $m$  is even, and we are asked whether there exists a partition of  $\mathcal{V} = \mathcal{S} \cup \mathcal{T}$  so that  $|\mathcal{S}| = |\mathcal{T}| = \frac{m}{2}$  and there is no edge from  $\mathcal{T}$  to  $\mathcal{S}$ . For any  $\text{ONEWAYBISECTION}$  instance, we construct a  $\text{MINBAYESIANLOSS}$  instance as follows.

The alternatives are the vertices. The preferences  $R$  are obtained by McGarvey’s trick (McGarvey, 1953) so that the positive edges in  $\text{WMG}(R)$  are the same as in  $G$ , and all positive weights are 2.  $\varphi = 2^{-m^2}$ .  $l = 1 - 2^{-m^2/4}$ .

For any  $A \subseteq \mathcal{C}$ , we let  $\Pr(A \succ \bar{A}|R)$  denote the posterior probability that all alternatives in  $A$  are preferred to all alternatives in  $\bar{A}$ . That is,  $h(A) = \sum_{W: A \succ_W \bar{A}} \Pr(W|R)$ . The next claim follows after calculations in (Elkind and Shah, 2014; Azari Soufiani et al., 2014).

**Claim 3.**  $\Pr(A \succ \bar{A}|R) = \prod_{a \in A, b \in \bar{A}} F(a, b)$ , where

$$F(a, b) = \begin{cases} \frac{1}{1+\varphi^2} & \text{if } w_R(a, b) = 2 \\ \frac{\varphi}{1+\varphi^2} & \text{if } w_R(a, b) = -2 \\ 1/2 & \text{if } w_R(a, b) = 0 \end{cases}$$

Therefore, if the  $\text{ONEWAYBISECTION}$  instance has a solution  $A$ , then the expected Bayesian loss for any alternative in  $A$  is at most  $1 - \Pr(A \succ \bar{A}|R) \leq 1 - 2^{-m^2/4}$ ,

which means that the MINBAYESIANLOSS instance is a “yes” instance. If the ONEWAYBISECTION instance does not have a solution, then for any alternative  $a \in \mathcal{C}$ , the expected Bayesian loss is at least  $1 - \binom{m}{m/2} \frac{\varphi^2}{1+\varphi^2} > 1 - 2^{m \log m} \frac{\varphi^2}{1+\varphi^2} > 1 - 2^{-m^2/4}$ , which means that the MINBAYESIANLOSS instance is a “no” instance.  $\square$

We now present the Markov chain  $\mathfrak{M}_C$  for Condorcet’s model in Algorithm 3, which runs  $\mathfrak{M}_C$  for  $N$  steps.  $\mathfrak{M}_C$  is an *independent sampler* and starts at an arbitrary state that maximizes the likelihood. In each step, a candidate next state is drawn independent of the current state, which means that  $p_X(\cdot)$  is the same for all  $X$ . We let  $p(\cdot)$  denote this proposal distribution.

In  $\mathfrak{M}_C$ ,  $p(\cdot)$  is the posterior probability *assuming that the prior is uniform*. In other words, for any  $W \in \mathcal{B}(\mathcal{C})$ ,  $p(W)$  is proportional to  $\Pr(R|W)$ . A binary relation in  $\mathcal{B}(\mathcal{C})$  can be efficiently generated from  $p(\cdot)$  by generating pairwise comparisons between alternatives independently, such that for each pair of alternatives  $(a, b)$ ,  $\frac{\Pr(a \succ_W b)}{\Pr(b \succ_W a)} = \frac{\varphi^{R[b \succ a]}}{\varphi^{R[a \succ b]}} = \varphi^{R[b \succ a] - R[a \succ b]}$ .

---

**Algorithm 3** Markov chain  $\mathfrak{M}_C$  for Condorcet’s model.

---

- 1: **Inputs:** a profile  $R$ , a prior  $\Pr_C$  over  $\mathcal{L}(\mathcal{C})$ , and the number of iterations  $N$ .
  - 2: Let  $V \in \mathcal{B}(\mathcal{C})$  denote a binary relation with the maximum likelihood computed by Proposition 2.
  - 3: **for**  $t=1$  **to**  $N$  **do**
  - 4:   Generate  $W \in \mathcal{B}(\mathcal{C})$  where all pairwise comparisons are generated independently such that for any  $(a, b)$ ,  $\frac{\Pr(a \succ b)}{\Pr(b \succ a)} = \varphi^{R[b \succ a] - R[a \succ b]}$ .
  - 5:   With probability  $\min\{\frac{\Pr_C(W)}{\Pr_C(V)}, 1\}$  let  $V = W$ .
  - 6: **end for**
  - 7: **return**  $V$ .
- 

**Theorem 5.** *The mixing time of  $\mathfrak{M}_C$  in Algorithm 3 is  $O((\ln \frac{d_{max}}{d_{max}-1})^{-1} (\ln d_{max} + m \ln m + \ln \epsilon^{-1}))$ .*

**Proof:** We apply a result by Liu (1996) to prove the upper bound on the variation distance.

**Lemma 6** ((Liu, 1996)). *For any independent sampler starting at  $V$ , we have*

$$\Delta_V(t) \leq \frac{(1 - \min_W \{p(W)/\pi(W)\})^t}{2\sqrt{\pi(V)}}$$

For any  $W$ , we have  $p(W) = \frac{\Pr(R|W)}{\sum_{U \in \mathcal{B}(\mathcal{C})} \Pr(R|U)}$  and  $\pi(W) = \Pr(W|R) = \frac{\Pr(R|W) \cdot \Pr(W)}{\sum_{U \in \mathcal{B}(\mathcal{C})} \Pr(R|U) \cdot \Pr(U)}$ . Therefore  $\frac{p(W)}{\pi(W)} = \frac{1}{\Pr(W)} \cdot \frac{\sum_{U \in \mathcal{B}(\mathcal{C})} \Pr(R|U) \cdot \Pr(U)}{\sum_{U \in \mathcal{B}(\mathcal{C})} \Pr(R|U)} \geq \min_U \{\frac{\Pr(U)}{\Pr(Y)}\} \geq \frac{1}{d_{max}}$ . By Lemma 6 we have  $\Delta_V(t) \leq \frac{1}{2\sqrt{\pi(V)}} \cdot (1 - \frac{1}{d_{max}})^t$ . Therefore,  $\tau_V(\epsilon)$  is  $O((\ln \frac{d_{max}}{d_{max}-1})^{-1} (\ln \pi(V)^{-1} + \ln \epsilon^{-1}))$ . When  $V$  maxi-

mizes the likelihood, we have  $\pi(V) \geq \frac{1}{d_{max} m!}$ . Applying Stirling’s formula we have  $\ln \pi(V)^{-1}$  is  $O(\ln d_{max} + m \ln m)$ , which proves the theorem.  $\square$

## 5 EXPERIMENTS

Most theoretical results in this paper are based on worst-case analysis. In this section we present some preliminary experimental results to illustrate the efficiency of our algorithms for real-world ranking data.

**Dataset:** We use the weighted majority graph dataset from Preflib (Mattei and Walsh, 2013) ([www.Preflib.org](http://www.Preflib.org)). Most of these datasets are collected from political elections. For each WMG, we normalize the weights to  $[-1, 1]$  by dividing all weights by the heaviest one. This is without loss of generality because we can set  $\varphi$  appropriately.

All experiments were run on a laptop with Intel i7-4600U processor, 8GB memory, and 256 GB SSD hard drive, running Windows 8.1 (64bit) and Python 2.7.9 (32bit).

### 5.1 MALLOWS’ MODEL

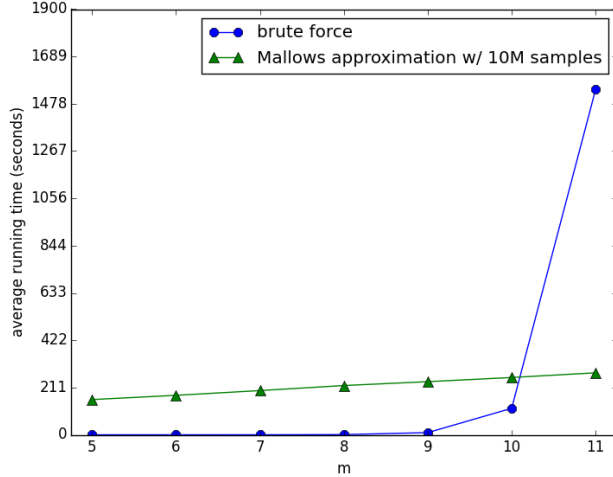
We tested Algorithm 1 with Algorithm 2 for the decision problem with Mallows’ model,  $\varphi = 0.9$ ,  $\mathcal{D} = \mathcal{C}$ , uniform prior, and the exact Top-1 loss function for  $m = 5$  through 11. We use brute-force enumeration to compute the optimal decision and discard the first 1/8 samples in our MCMC algorithm as burn-in. The average running time and convergence of Bayesian loss computed for all datasets with the same number of alternatives are shown in Figure 1.<sup>4</sup>

In Figure 1 (a) we observe that the running time of brute-force search grows exponentially in  $m$  (because the number of parameters is  $m!$ ) while the running time for our MCMC algorithm grows linearly in  $m$ . Figure 1 (b) shows the reduction of the total difference between the estimated Bayesian loss via MCMC and the ground truth computed by brute-force search w.r.t. the number of samples. We note that this is *not* the variance distance. The average is taken over all datasets with the same number of alternatives. We observe that the total difference can be effectively reduced by increasing the number of samples. Moreover, when we use 10 million samples, the optimal Bayesian decision is correct in 109 out of 115 datasets ( $\approx 95\%$ ) as shown in Table 2.

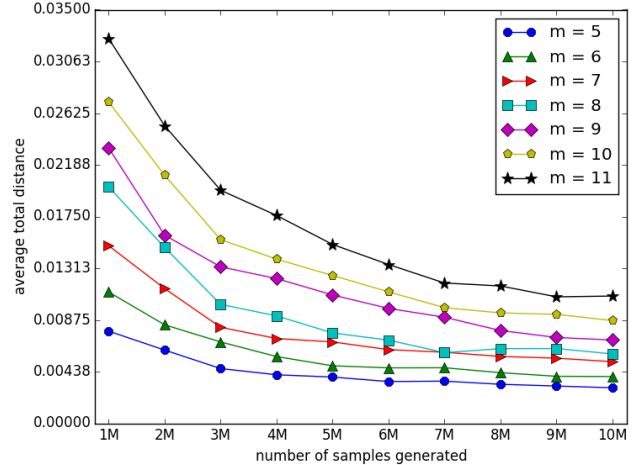
| $m =$     | 5  | 6  | 7  | 8  | 9  | 10 | 11 | Total |
|-----------|----|----|----|----|----|----|----|-------|
| correct   | 18 | 13 | 18 | 11 | 17 | 20 | 12 | 109   |
| incorrect | 0  | 0  | 2  | 1  | 3  | 0  | 0  | 6     |

Table 2: The number of correct and incorrect Bayesian decisions for Mallows’ model.

<sup>4</sup>We have also tested the efficiency of the algorithm with larger  $N$  in Algorithm 2, and observe that the efficiency is in general lower than the efficiency when all samples are used.



(a) Average running time.



(b) Average total difference.

Figure 1: The average running time and average total difference of our algorithm for Mallows' model.

## 5.2 CONDORCET'S MODEL

We tested Algorithm 1 with Algorithm 3 for the decision problem with Condorcet's model,  $\varphi = 0.9$ ,  $\mathcal{D} = \mathcal{C}$ , uniform prior, and the exact Top- $\lfloor \frac{m}{2} \rfloor$  loss function for  $m = 5$  through 11. Given a binary relation  $W \in \mathcal{B}(\mathcal{C})$  and an alternative  $d$ , the exact Top- $k$  loss can be computed by the following polynomial-time algorithm. We say an alternative  $a$  dominates another alternative  $b$  in  $W$ , if there is a directed path from  $a$  to  $b$  in  $W$ . For each alternative  $c$ , we first compute the set of all alternatives that dominate  $c$ , denoted by  $D_c$ . By definition we have  $c \in D_c$ . Then, the loss of  $d$  is 0 if and only if there exists an alternative  $c$  such that (1)  $|D_c| = k$  and (2)  $d \in D_c$ ; otherwise the loss of  $d$  is 1.

By Theorem 4, when  $k$  is small there exists a polynomial-time algorithm to compute the Bayesian losses. This makes experiments on Preflib data hard because the algorithm can efficiently compute the optimal Bayesian losses for reasonably large  $m$  (for example  $m = 20$ ), and there are not enough datasets with  $m > 20$ . Therefore, we only show the reduction in average total difference in Figure 2. Extensive experimental studies on real-world datasets are left for future work.

In Figure 2 we observe that (i) for the same number of samples the total difference for Condorcet's model is smaller than the total difference for Mallows' model, and (ii) with the same number of samples, larger  $m$  corresponds to smaller total difference. (The total difference for  $m = 11$  is too small to be seen clearly in Figure 2.) This may be due to two reasons. First,  $\mathfrak{M}_C$  has a better theoretical guarantee (that it is fast mixing) than  $\mathfrak{M}_M$ . Second, for Condorcet's model the probability for the loss of any decision (alternative) to be 0 w.r.t. the exact Top- $\lfloor \frac{m}{2} \rfloor$  loss function is small, which means that for most generated samples the loss of all alternatives is 1.

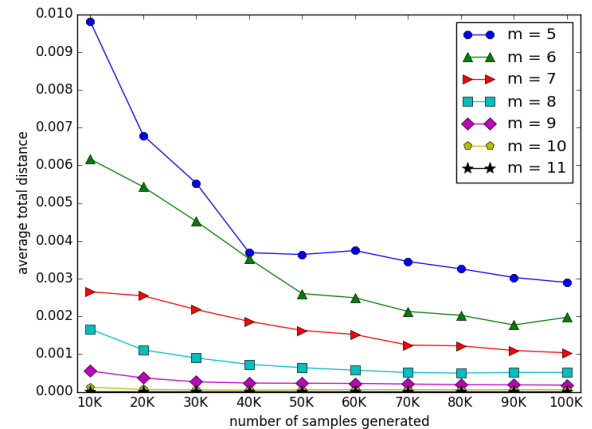


Figure 2: The average total difference for Condorcet's model.

## 6 SUMMARY AND FUTURE WORK

We have proposed and analyzed two MCMC algorithms for making optimal Bayesian decisions for two popular ranking models w.r.t. any prior and loss function. There are many open questions and future directions. Can we improve the analysis to show that the Markov chain for Mallows' model is rapid mixing or prove that the Bayesian decision problems are hard to approximate by efficient randomized algorithms? Can we design and analyze other Markov chain samplers? How to further improve the performance of the Markov chain sampler in practice? How does the Markov chain approach compare to other popular statistical and machine learning techniques, for example importance sampling?

## ACKNOWLEDGMENTS

We thank Zhibing Zhao and anonymous reviewers of UAI-15 for helpful comments. This work is supported in part by NSF CAREER under award number IIS-1453542.

## References

- Nir Ailon, Moses Charikar, and Alantha Newman. Aggregating inconsistent information: Ranking and clustering. In *Proc. STOC*, pages 684–693, 2005.
- Alnur Ali and Marina Meila. Experiments with Kemeny ranking: What works when? *Mathematical Social Sciences*, 64(1):28–40, 2012.
- Hossein Azari Soufiani, David C. Parkes, and Lirong Xia. Random utility theory for social choice. In *Proc. NIPS*, pages 126–134, 2012.
- Hossein Azari Soufiani, William Chen, David C. Parkes, and Lirong Xia. Generalized method-of-moments for rank aggregation. In *Proc. NIPS*, 2013a.
- Hossein Azari Soufiani, Hansheng Diao, Zhenyu Lai, and David C Parkes. Generalized random utility models with multiple types. In *Proc. NIPS*, 2013b.
- Hossein Azari Soufiani, David C. Parkes, and Lirong Xia. Statistical decision theory approaches to social choice. In *Proc. NIPS*, 2014.
- John Bartholdi, III, Craig Tovey, and Michael Trick. Voting schemes for which it can be difficult to tell who won the election. *Social Choice and Welfare*, 6:157–165, 1989.
- Nadja Betzler, Michael R. Fellows, Jiong Guo, Rolf Niedermeier, and Frances A. Rosamond. Fixed-Parameter Algorithms for Kemeny Scores. In *Algorithmic Aspects in Information and Management*, volume 5034 of *Lecture Notes in Computer Science*, pages 60–71, 2008.
- Marquis de Condorcet. *Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix*. Paris: L'Imprimerie Royale, 1785.
- Persi Diaconis and Phil Hanlon. Eigenanalysis for some examples of the Metropolis algorithm. *Contemporary Mathematics*, 138: 99–117, 1992.
- Cynthia Dwork, Ravi Kumar, Moni Naor, and D. Sivakumar. Rank aggregation methods for the web. In *Proc. WWW*, pages 613–622, 2001.
- Edith Elkind and Nisarg Shah. How to Pick the Best Alternative Given Noisy Cyclic Preferences? In *Proc. UAI*, 2014.
- Uriel Feige and Orly Yahalom. On the Complexity of Finding Balanced Oneway Cuts. *Information Processing Letters*, 87(1):1–5, 2003.
- Sumit Ghosh, Manisha Mundhe, Karina Hernandez, and Sandip Sen. Voting for movies: the anatomy of a recommender system. In *Proc. AGENTS*, pages 434–435, 1999.
- W. Keith Hastings. Monte Carlo Sampling Methods Using Markov Chains and Their Applications. *Biometrika*, 57(1):97–109, 1970.
- Edith Hemaspaandra, Holger Spakowski, and Jörg Vogel. The complexity of Kemeny elections. *Theoretical Computer Science*, 349(3):382–391, December 2005.
- Mark Jerrum and Alistair Sinclair. The Markov chain Monte Carlo method: an approach to approximate counting and integration. In Dorit S. Hochbaum, editor, *Approximation algorithms for NP-hard problems*, pages 482–519. PWS Publishing Company, 1996.
- Claire Kenyon-Mathieu and Warren Schudy. How to Rank with Few Errors: A PTAS for Weighted Feedback Arc Set on Tournaments. In *Proc. STOC*, pages 95–103, 2007.
- Jen-Wei Kuo, Pu-Jen Cheng, and Hsin-Min Wang. Learning to Rank from Bayesian Decision Inference. In *Proc. CIKM*, pages 827–836, 2009.
- Jun S. Liu. Metropolized independent sampling with comparisons to rejection sampling and importance sampling. *Statistics and Computing*, 6(2):113–119, 1996.
- Tie-Yan Liu. *Learning to Rank for Information Retrieval*. Springer, 2011.
- Bo Long, Olivier Chapelle, Ya Zhang, Yi Chang, Zhaohui Zheng, and Belle Tseng. Active Learning for Ranking Through Expected Loss Optimization. In *Proc. SIGIR*, pages 267–274, 2010.
- Tyler Lu and Craig Boutilier. Learning mallows models with pairwise preferences. In *Proc. ICML*, pages 145–152, 2011.
- Colin L. Mallows. Non-null ranking model. *Biometrika*, 44(1/2): 114–130, 1957.
- Andrew Mao, Ariel D. Procaccia, and Yiling Chen. Better human computation through principled voting. In *Proc. AAAI*, 2013.
- Nicholas Mattei and Toby Walsh. PrefLib: A Library of Preference Data. In *Proc. ADT*, 2013.
- David C. McGarvey. A theorem on the construction of voting paradoxes. *Econometrica*, 21(4):608–610, 1953.
- Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of State Calculations by Fast Computing Machines. *Journal of Chemical Physics*, 21(6):1087–1092, 1953.
- Sahand Negahban, Sewoong Oh, and Devavrat Shah. Iterative ranking from pair-wise comparisons. In *Proc. NIPS*, pages 2483–2491, 2012.
- Daniele Porello and Ulle Endriss. Ontology Merging as Social Choice: Judgment Aggregation under the Open World Assumption. *Journal of Logic and Computation*, 2013.
- Ariel D. Procaccia, Sashank J. Reddi, and Nisarg Shah. A maximum likelihood approach for selecting sets of alternatives. In *Proc. UAI*, 2012.
- Karthik Raman and Thorsten Joachims. Methods for Ordinal Peer Grading. In *Proc. SIGKDD*, pages 1037–1046, 2014.
- Karthik Raman and Thorsten Joachims. Bayesian Ordinal Peer Grading. In *Proc. L@S*, 2015.
- Alistair Sinclair. Improved Bounds for Mixing Rates of Markov Chains and Multicommodity Flow. *Combinatorics, Probability and Computing*, 1(4):351–370, 1992.
- Alistair Sinclair and Mark Jerrum. Approximate counting, uniform generation and rapidly mixing Markov chains. *Information and Computation*, 82(1):93–133, 1989.
- A. F. M. Smith and G. O. Roberts. Bayesian Computation Via the Gibbs Sampler and Related Markov Chain Monte Carlo Methods. *Journal of the Royal Statistical Society. Series B*, 55(1):3–23, 1993.
- H. Peyton Young. Condorcet's theory of voting. *American Political Science Review*, 82:1231–1244, 1988.

---

# Do-calculus when the True Graph Is Unknown

---

**Antti Hyttinen**

HIIT, Dept. Computer Science  
University of Helsinki  
Finland

**Frederick Eberhardt**

Humanities and Social Sciences  
California Institute of Technology  
Pasadena, CA, USA

**Matti Järvisalo**

HIIT, Dept. Computer Science  
University of Helsinki  
Finland

## Abstract

One of the basic tasks of causal discovery is to estimate the causal effect of some set of variables on another given a statistical data set. In this article we bridge the gap between causal structure discovery and the *do*-calculus by proposing a method for the identification of causal effects on the basis of arbitrary (equivalence) classes of semi-Markovian causal models. The approach uses a general logical representation of the equivalence class of graphs obtained from a causal structure discovery algorithm, the properties of which can then be queried by procedures implementing the *do*-calculus inference for causal effects. We show that the method is more efficient than determining causal effects using a naive enumeration of graphs in the equivalence class. Moreover, the method is complete with respect to the identifiability of causal effects for settings, in which extant methods that do not require knowledge of the true graph, offer only incomplete results. The method is entirely modular and easily adapted for different background settings.

## 1 INTRODUCTION

The theory of causal learning is aimed at finding ways to estimate causal effects in a variety of different settings. In the most basic setting the starting point is a statistical data set of measurements over the variables of interest. In this article we explore how quantitative causal effects can be estimated from such data alone, that is, without the additional knowledge of the causal structure.

When the true causal structure (the causal graph) *is* known, the well-known *do*-calculus enables the complete inference (i.e., the identification) of causal effects from the passive observational distribution over the variables (Pearl, 2000;

Shpitser and Pearl, 2006b). However, full knowledge of the true graph requires a rather extensive understanding of the system under investigation. Data alone is in general insufficient to uniquely determine the true causal graph. Even complete discovery methods will usually leave the graph underdetermined (Spirtes et al., 1993).

Here we develop a general method for the combined task of causal structure discovery and the inference about causal effects.<sup>1</sup> Leveraging a constraint satisfaction approach to connect the output of causal discovery algorithms to the *do*-calculus, our method enables the identification of causal effects for arbitrary (equivalence) classes of semi-Markovian causal models (DAGs with latent variables). The primary advantages of the approach are that (i) it does not assume a unique true causal structure, (ii) it is not restricted to particular types of equivalence classes of causal structures, such as partial ancestral graphs (PAGs), (iii) it provides an algorithm that outputs (when possible) at least one estimator of the causal effect, rather than only specifying rules of a calculus, (iv) it considers all *do*-calculus inferences, not just e.g. the so-called backdoor conditions, and (v) it gives the user flexibility e.g. in how statistical conflicts in the data are handled and how the possibility of multiple estimators of a causal effect is addressed. To simplify notation we will, throughout this article, describe our method using a single observational data set as input. However, we emphasize that the method is extremely general, so, where relevant, we will indicate how the approach is extended or adapted to other scenarios.

Figure 1 gives an overview of the computational flow of the proposed method. The method determines whether a causal effect of the form  $P(y \mid do(x), w)$  is identified given a data set as input, and if so, provides a numerical estimate. We use a (complete) causal discovery method to extract from the data as much information as possible about the true causal graph in terms of so-called *d-separation/connection constraints*. We encode these constraints in the language of propositional logic for the constraint solving component, thereby *implicitly* representing the equivalence class

---

<sup>1</sup>See Section 5 for a discussion of related approaches.

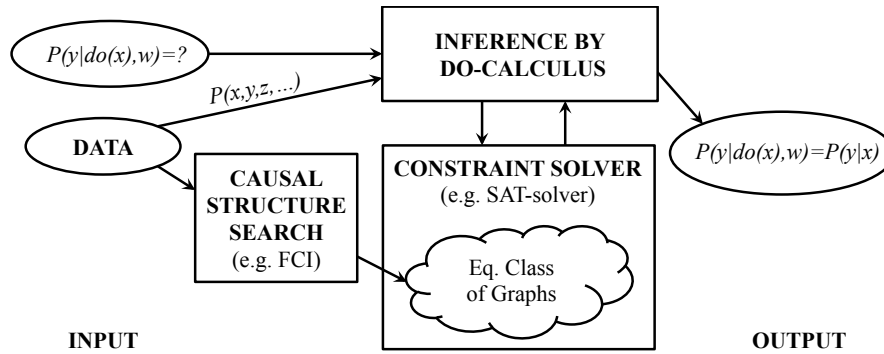


Figure 1: Overall structure of the system.

of causal structures (Hyttinen et al., 2013, 2014). This enables the application of modern constraint solving techniques, such as Boolean satisfiability (SAT) solvers (Biere et al., 2009). We can then use the *do*-calculus to determine whether the causal effect  $P(y | do(x), w)$  is identified by alternating between *do*-calculus inferences and constraining the equivalence class to graphs for which no estimator of the causal effect has been found yet. If the causal effect is identifiable, we obtain a formula and a numerical estimate of the causal effect from the joint probability distribution over the variables.

Our method enables considerable flexibility in addressing the identification problem: The representation of the candidate causal structures in terms of a logical formula frees us from the restriction to settings where standard graphical representations of equivalence classes of causal graphs apply. We can include a wide variety of background constraints or additional knowledge, e.g., from experiments. We can leverage the full inferential power of the *do*-calculus without having to explicitly enumerate every causal structure consistent with the data. Nevertheless, we can (and do) instantiate an implicit exhaustive search that ensures that we preserve the completeness guarantees of *both* the causal discovery procedure and the *do*-calculus.

The paper is structured as follows: In Section 2 we describe the model space and assumptions used and give a concise problem statement. Section 3 explains the inference algorithms, whose completeness properties are discussed in Section 4. In Section 5 we describe known results and approaches that are closely related to ours or that provide context. Section 6 provides simulated results illustrating our main points.

## 2 PROBLEM SETUP

Following the standard set-up of the *do*-calculus, we assume that the causal system can be represented by a semi-Markovian causal model (SMCM). In other words, the un-

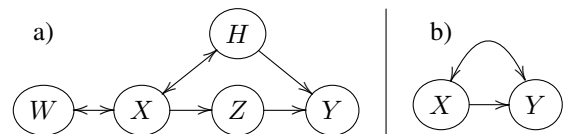


Figure 2: a) Example of a SMCM graph for which the causal effect  $P(y|do(x))$  is identifiable even when the graph is unknown. b) Example of a SMCM graph for which the causal effect  $P(y|do(x))$  is not identifiable even when the graph is known (since the graph includes a *hedge*).

derlying causal structure over a set of causal variables  $\mathbf{V}$  is described by a directed acyclic graph  $G$ , in which the directed edges correspond to direct causal relations between the variables (relative to  $\mathbf{V}$ ), and confounding of any two observed variables by some unobserved common cause  $U$  is represented by a bi-directed edge between the variables (thereby omitting  $U$  for simplicity in the graph; see Figure 2). The causal structure gives rise to a probability distribution that is assumed to be Markov and faithful to the graph. No further parametric assumption about the distribution is made. Importantly, Markov and faithfulness ensure that the probabilistic (in)dependencies of the distribution correspond to *d*-separation/connection relations in the causal graph.<sup>2</sup>

Under specific *d*-separation conditions on the underlying causal structure, the rules of the *do*-calculus (see Figure 3) license inferences between the passive observational distribution  $P(\mathbf{V})$  and the corresponding (conditional) interventional distributions  $P(y | w, do(x))$ , where one or more variables  $x \subset \mathbf{V}$  have been subject to intervention and variables  $w \subset \mathbf{V}$  are conditioned on. By using an additional exogenous intervention variable  $I_X$  with  $I_X \rightarrow X$  for each variable  $X$ , the *d*-separation conditions of the *do*-calculus can be stated as in Figure 3 (see Pearl (1995, p. 686) for the

<sup>2</sup>See Spirtes et al. (1993) for a precise statement of the assumptions and for a definition of *d*-separation.

**Rule 1** (Insertion/deletion of observations):

$$P(y|do(x), z, w) = P(y|do(x), w) \text{ if } Y \perp\!\!\!\perp Z|X, W||X$$

**Rule 2** (Action/observation exchange):

$$P(y|do(x), do(z), w) = P(y|do(x), z, w) \text{ if } Y \perp\!\!\!\perp I_Z|X, Z, W||X$$

**Rule 3** (Insertion/deletion of actions):

$$P(y|do(x), do(z), w) = P(y|do(x), w) \text{ if } Y \perp\!\!\!\perp I_Z|X, W||X$$

**Rule 4** (Marginalization/sum-rule):

$$P(y|do(x), w) = \sum_z P(y, z|do(x), w)$$

**Rule 5** (Conditioning):

$$P(y|do(x), z, w) = \frac{P(y, z|do(x), w)}{\sum_y P(y, z|do(x), w)}$$

**Rule 6** (product/chain-rule):

$$P(y, z|do(x), w) = P(y|do(x), w, z)P(z|do(x), w)$$

Figure 3: Rules of the do-calculus.

proof of equivalence to the standard conditions; see also Spirtes et al. (1993, p. 79)). The d-separation conditions of each rule have the general form of ‘ $Y \perp\!\!\!\perp Z | X, W || X$ ’, where  $W, X, Y, Z$  are disjoint sets of variables in the graph (including intervention variables), and ‘ $|| X$ ’ denotes an intervention on  $X$ : any edges with arrowheads into the variables in  $X$  are cut.

Given a graph  $G$ , the identifiability of a causal effect is now defined as follows (see Pearl (2000), Def. 3.2.4, p. 77): a causal effect is identifiable if and only if it can be uniquely computed from  $G$  and any positive input distribution  $P()$  that is Markov to  $G$ , i.e., there are no two causal models with structure  $G$  that are Markov to  $P()$  but have different numerical values for the causal effect.

An algorithm to apply the do-calculus when the true graph is known was developed by Tian and Pearl (2002), which with some modifications was shown to be complete for the identification of (conditional) causal effects by Shpitser and Pearl (2006b) (see also Huang and Valtorta (2006)). The *Shpitser algorithm* provides (given the graph and the observational distribution) one estimator of the causal effect, if such an estimator exists. When the causal effect is non-identifiable, it returns a feature of the graph, known as a *hedge*, that proves non-identifiability (see Figure 2b for an example; see Shpitser and Pearl (2006b) for the exact definition).

Since we do not assume that the true graph  $G$  is known, we take a causal effect to be identifiable given the equivalence class of causal structures deemed consistent with the input data if and only if the causal effect is (Pearl-) identifiable by the same estimator for each member of the equivalence class.

As shown in the simulations in Section 6, the causal effect is very often not uniquely identifiable from data when the true graph is unknown. So, instead of outputting only whether an effect is identifiable, and the estimate if it is, we follow Maathuis et al. (2009), who output a (multi)set of causal effect estimates that in some cases can be used to obtain bounds on the true causal effect. This leads to the following problem statement.

### Problem Statement

**INPUT:** Data set  $D$  generated from an SMCM over variables  $\mathbf{V}$  and a query about a causal effect  $P(y | do(x), w)$ .

**TASK:** Output a set of causal effect estimates  $S$  such that it includes an estimate for  $P(y | do(x), w)$  for any causal structure that is consistent with  $D$ . Include ‘NA’ in  $S$ , if the causal effect is not identifiable for some causal structure consistent with  $D$ .

## 3 THE APPROACH

We proceed by describing the main contribution of this work: a general method for the estimation of causal effects. In the following, we will specify the main components (recall Figure 1) of our approach. First, we give details on how we connect the causal structure discovery algorithm of choice with the constraint solving component that maintains a logical representation of the equivalence class of models under consideration. Then, we describe the do-calculus inference component and its iterative interactions with the constraint solver.

### 3.1 Querying the Equivalence Class

We use (for purposes of illustration) the FCI-algorithm (Spirtes et al., 1993) to determine the equivalence class of candidate causal structures from the data set. The FCI-algorithm considers the same class of causal models as the do-calculus: acyclic causal structures with latent variables. It is complete with respect to knowledge about the underlying causal structure that can be obtained from conditional independence tests. Most importantly, FCI achieves this d-separation completeness while performing very few redundant tests. It thus also lends itself to the efficient characterization of the equivalence class in terms of a small set of d-separation constraints that can be fed to the constraint solver.

---

**Algorithm 1** Do-calculus Inference.

---

Input:  $P(y|do(x), w)$ , an equivalence class  $E$  of SMCM graphs.

Initialize the set  $S$  of causal effect estimates as empty.

While  $E$  is nonempty:

Find a graph  $G$  from the equivalence class  $E$ .

Find a formula  $F$  by calling Shpitser’s algorithm for graph  $G$ . If the algorithm does not find a formula but returns a hedge  $H$ , restrict eq. class  $E$  not to include  $H$ , add NA to  $S$  and continue the loop from the beginning.

Find a derivation for  $F$  by calling Algorithm 2 for graph  $G$  and the input distributions used in  $F$ .

Using  $F$  and the (estimated)  $P(\mathbf{V})$ , compute the estimate for the causal effect and add it to  $S$ .

Restrict the eq. class  $E$  to not satisfy at least one of the required d-separations in derivation  $D$ .

Return a set  $S$  of numerical causal effect estimates.

---

We translate the d-separation constraints of the equivalence class into a logical representation using the ASP-encoding of Hyttinen et al. (2014). We can then query the constraint solver to obtain graphs from the equivalence class or to check whether any graphical conditions, such as d-separations or ancestral relations, apply to all, some or none of the members in the equivalence class, and we can further restrict the equivalence class with additional constraints.

For other settings or different background assumptions, the FCI algorithm can be substituted with any other structure discovery method. If one has reason to think that there are no latent variables, we would recommend using an exact Bayesian search algorithm, or the PC- or GES-algorithms if something more scalable is required (Spirtes et al., 1993; Chickering, 2002). If the causal constraints are to be obtained from heterogeneous data sets, possibly including experimental data or background knowledge, then a search algorithm such as GIES, IOD or a SAT-based procedure may be better (Hauser and Bühlmann, 2012; Tillman and Spirtes, 2011; Triantafillou and Tsamardinos, 2014; Hyttinen et al., 2014). The overall completeness of our method depends in part, of course, on whether the causal discovery method is complete.

### 3.2 Identifying Causal Effects

Algorithm 1 instantiates the *do*-calculus inference on a given equivalence class. It queries the constraint solver for a graph  $G$  in the equivalence class (one truth-value assignment to the logical formula) and calls Shpitser’s algorithm on  $G$  to identify the desired causal effect.

If Shpitser’s algorithm fails to identify the effect and returns a hedge  $H$ , then the causal effect is non-identifiable for  $G$ . Consequently, Algorithm 1 adds ‘NA’ to the set

---

**Algorithm 2** Do-calculus Derivation.

---

Input:  $P(y|do(x), w)$ , a SMCM graph  $G$ , and a set of distributions  $P = \{P_1, \dots\}$ .

For each  $P_i$  in  $P$ :

Derive the distributions computable from  $P_i$  using the rules of the do-calculus such that:

- The required d-separation conditions are satisfied by  $G$ .
- All variables appearing in the derived distributions are ancestors of  $Y \cup W$  (see Shpitser and Pearl (2006a)).
- For an application of the product rule, both required distributions are in  $P$ .

Add the new distributions to  $P$  and record the used rules and the required d-separations.

If  $P(y|do(x), w)$  was derived, return the formula, the rules, and the d-separations used on the way.

Return “the effect is not identifiable”.

---

$S$  of causal effect estimates to mark the non-identifiability. In addition, we restrict the equivalence class to not include any graphs that have the hedge  $H$ , as the causal effect is unidentifiable for such graphs as well.

If Shpitser’s algorithm returns a formula  $F$ , then the causal effect is identifiable on the basis of the (marginal conditional) distributions  $\{P_1, \dots\}$  used in  $F$ . Algorithm 2 is then called with the causal effect query, the graph  $G$  and the list of distributions  $\{P_1, \dots\}$  to obtain a ‘derivation’ for the formula. This derivation specifies the rules of the do-calculus used to derive the formula  $F$  and consequently the set of d-separation constraints  $C$  that warrant the use of this estimate. The numerical estimate is added to  $S$ . This estimate is now valid for all graphs that satisfy the d-separation constraints in  $C$ . Then, the equivalence class is again restricted to disregard such graphs by ensuring that at least one of the constraints in  $C$  is no longer satisfied, i.e., we add the negation of the conjunction of constraints in  $C$  to the constraint solver. We repeatedly solve for a new graph  $G$  from the restricted equivalence class until the class becomes empty, at which time we have the solution to the Problem Statement. Note that the repeated restrictions of the equivalence class avoid an explicit enumeration of all the members of the equivalence class, allowing for faster operation.

Algorithm 2 implements the search for a valid do-calculus derivation for a formula. We need such a derivation as Shpitser’s algorithm does not output the set of d-separations needed for the validity of the formula. Algorithm 2 does an exhaustive breadth-first-search, producing computable distributions that are warranted by the input graph and do-calculus. It stops when a derivation for the causal effect is found. The algorithm can be made sufficiently efficient because 1) we only input the distributions that are used in



the formula given by Shpitser’s algorithm, 2) we use the fact that only variables that are ancestors of  $Y \cup W$  can be helpful in determining the causal effect (Shpitser and Pearl, 2006b), and 3) causal structures usually permit the identification of fairly few distributions.

Figure 9 at the end of the paper shows an example run of Algorithm 1 for one particular equivalence class.

## 4 COMPLETENESS RESULTS

The completeness properties of our method are derivative of the completeness properties of the causal discovery algorithm of choice and the *do*-calculus. Given a complete structure search algorithm, such as FCI, we obtain in the large sample limit the Markov equivalence class of the true causal structure. If the causal effect in this equivalence class is non-identifiable, it is either because one graph in the equivalence class contains a *hedge*, which proves non-identifiability, or because there are two graphs which have different estimators for the causal effect. Algorithm 1 repeatedly performs the (complete) Shpitser algorithm on members of the equivalence class, each time restricting the equivalence class to graphs for which the discovered derivations of an estimator do not hold. Consequently, its output must eventually identify a graph with a hedge if there is one, since implicitly the entire set of graphs in the equivalence class is enumerated. If no graph with a hedge is found, then Algorithm 1 only terminates once there is a derivation of an estimator of the causal effect for each graph in the equivalence class. A check whether these formulas are the same determines the identifiability. (In the presented version of the algorithms we only output the numerical estimates, but the formulas could easily be added to avoid any formal concern that there could be coincidentally identical numerical values of the causal effect derived from two different estimators.)

We note that the FCI algorithm is not complete with regard to so-called *Verma constraints* that can further restrict the equivalence class of causal structures (Shpitser and Pearl, 2008). We are not aware of any search algorithm that is complete in this regard. However, any specific Verma constraint that may be established for a particular case, can easily be included and the set of estimators our method returns will be complete with regard to that additional constraint.

For settings involving multiple data sets or experimental data sets, there exist d-separation complete structure search algorithms, but it is not known whether the *do*-calculus is complete for these settings. Certainly, Shpitser’s algorithm is restricted to the passive observational distribution. For the restricted experimental settings described in Bareinboim and Pearl (2012), we could replace Shpitser’s algorithm with Bareinboim’s method in Algorithm 1 and retain completeness, since Bareinboim shows completeness for the identification problem in these so-called “surrogate

experiments”. In the future, we hope our approach can aid the identification of causal effects from multiple data sets of non-identical populations (Bareinboim and Pearl, 2013).

## 5 RELATED WORK

Building the connection between causal structure discovery and causal effect inference seems essential if one wants to complete the aim of causal learning from data sets to causal effects. We consider it all the more important given that significant parts of the causal literature regard the problem of identifying the causal effect *given* the causal structure as entirely separate from the problem of discovering the causal structure in the first place. For example, the entire literature on algorithms applying the *do*-calculus assumes — generally without further discussion — that the causal graph is known (Tian and Pearl, 2002; Huang and Valtorta, 2006; Shpitser and Pearl, 2006b; Bareinboim and Pearl, 2012). In the general model space that the *do*-calculus allows for, the causal structure can hardly ever be uniquely determined from the passive observational distribution or even from the experimental distributions that Bareinboim and Pearl (2012) consider. Still, the algorithms rely on being able to check complicated features of the causal structure. Similarly, the methods of causal structure discovery often remain silent on how exactly one should determine the causal effects given their output equivalence class.<sup>3</sup> There need not be any harm in this division of labor if there is an obvious and satisfactory answer of how to connect the two. We assume that the standard proposal would be to take the equivalence class of causal structures output by a search algorithm, enumerate each member of the equivalence class, and perform the *do*-calculus algorithm on each member to determine (the identifiability of) the causal effect. Effectively, this is what is done in the IDA-algorithm which returns (multi-sets of) estimates of the causal effects of variables from an equivalence class of causal structures under the assumption that there are no latent variables (Maathuis et al., 2009). However, such an explicit enumeration of the members of an equivalence class can very quickly become unwieldy (see also (Malinsky, 2015)). In our simulations (Section 6) we show that our approach is more efficient than a naive enumeration combined with the *do*-calculus inference.

Zhang (2008), instead, developed a *do*-calculus directly for the equivalence classes represented by partial ancestral graphs (PAGs; see also Richardson and Spirtes (2003)). As he explains, the calculus is not complete and no inference algorithm to apply the calculus is given (although Zhang notes that his results could be used to improve on the ear-

<sup>3</sup>The same is not true for methods of causal discovery that include a parametric assumption (e.g. linearity, additive noise, non-Gaussianity, etc), since in these cases the identification of the qualitative causal effect generally corresponds to providing a quantitative estimate of it.

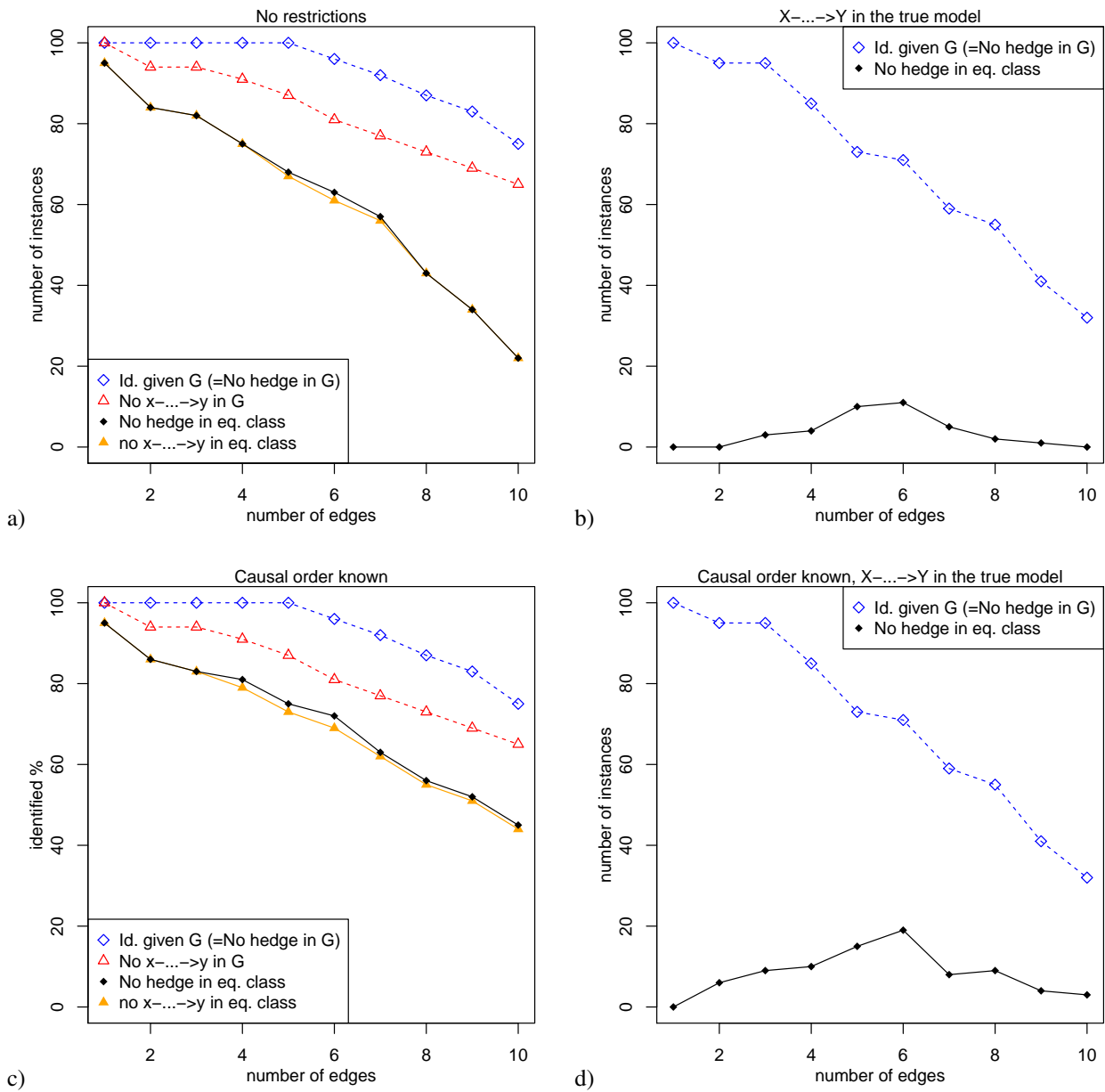


Figure 4: Identifiability of the causal effect  $P(y|do(x))$  over random 5-variable graphs.

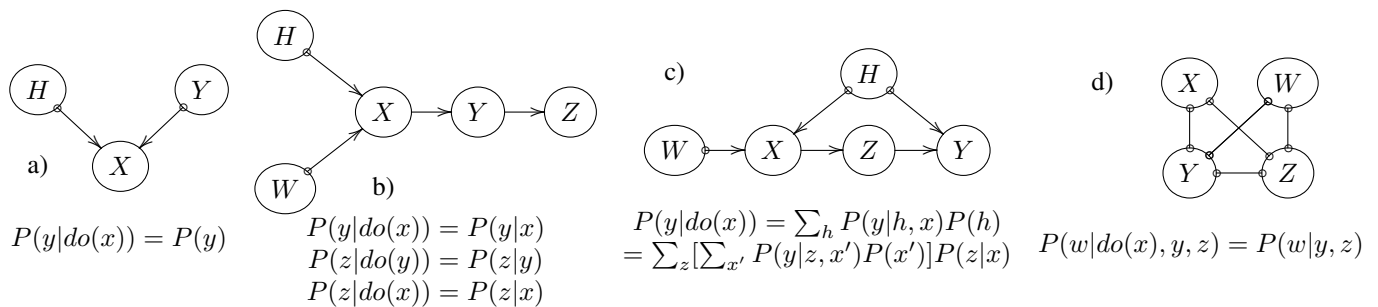


Figure 5: Some examples of graph equivalence classes where causal effects can be identified. The circles on the edges can be edge heads or tails (or both), as long as no new unshielded colliders are formed.

lier Prediction Algorithm by Spirtes et al. (1993)). His approach is closely related to ours, with the main difference that we do not restrict ourselves to equivalence classes of SMCMs that are PAGs and that our procedure is complete for his setting.

Also relying on PAGs, Maathuis and Colombo (2015) focus on a subset of Zhang’s invariance principles (which themselves are implied by the *do*-calculus) in order to specify conditions on a PAG that allow for the identification of an *adjustment set*, i.e., a set of variables  $W$  that block all so-called “backdoor paths” between  $X$  and  $Y$ , and allow for the causal effect  $P(y | do(x))$  to be estimated using  $P(y | w, x)$ . In our approach their “generalized backdoor criterion” corresponds to the simple and intuitive d-separation conditions that are required for the estimation of the causal effect ( $w \perp\!\!\!\perp I_x$  and  $y \perp\!\!\!\perp I_x | x, w$ ). These we can directly query on *any* class of SMCMs, not just on PAGs (or MAGs or DAGs).

More data-driven methods to find adjustment sets have also been developed (De Luna et al., 2011; Entner et al., 2013; VanderWeele and Shpitser, 2011). These do not rely explicitly on a graphical representation of the causal knowledge, but specify independence conditions that can be directly checked in the data. This is a very attractive direction of research, since it cuts out the graph from the inference procedure altogether. However, extant methods rely on a variety of general background assumptions about how the causal variables may be related (e.g. order assumptions) that we do not require. More generally, methods for the identification of adjustment sets obviously do not exhaust the identifiability conditions for causal effects for which the *do*-calculus was shown to be complete. Most prominently, the so-called “front-door” criterion for identifiability is not considered. Thus, one of the contributions of our method is to enable the full identification power of the *do*-calculus in settings when the causal structure is underdetermined.

## 6 SIMULATIONS

These simulations explore the identifiability of the causal effects when the true causal graph is unknown, the scalability of the methods presented in this paper, and, finally, the accuracy of different causal effect estimates when multiple estimators can be calculated. We implemented the algorithms using R with various packages (Tikka, 2014; Kalisch et al., 2012). Following Hyttinen et al. (2014) for the implementation of the constraint solving component, we employed the off-the-shelf state-of-the-art answer set programming (ASP) solver Clingo version 4.4.0, which at its core uses modern SAT solving techniques to perform a complete search for solutions, and at the same time allows for a natural high-level representation of the structural constraints in logical form (Gebser et al., 2011).

Figure 4 compares the identifiability of the basic causal ef-

fect  $P(y|do(x))$  when the graph is known vs. when only its equivalence class is known. Figure 4a shows the number of identified causal effects in random 5-variable graphs without any restrictions, as density increases. When only the equivalence class is known, the causal effect is almost always either trivially identified as  $X$  is discovered not to be an ancestor of  $Y$  in any member of the equivalence class, or trivially unidentifiable due to a possible hedge in the equivalence class. When the graph is known, a significant number of causal effects are identified even when  $X$  is an ancestor of  $Y$ . This is further highlighted in Figure 4b where  $X$  is required to be an ancestor of  $Y$  in the true graph. Almost no causal effects are identified when only the equivalence class is known. Only in very rare cases can one orient enough edges to deduce the absence of hedges from the equivalence class. This seems to happen when 4-7 edges are present; additional edges often prevent the determination of the orientation. Figure 4c and 4d consider the same comparison with the modification that the true causal order is known by the causal discovery algorithm, and thus fixed in the equivalence class. A few more effects are identified, but since the fixed causal order does not prevent bidirected edges, the improvement on non-trivial instances is limited in Figure 4d.

Figure 5 shows equivalence classes of graphs for which the causal effect is identified even when the true graph is unknown. In each case, we are able to deduce enough edge orientations to prevent the presence of a hedge, and to fix the orientation of the paths from  $X$  to  $Y$ . Note that Figure 5d shows an example for which the *do*-calculus formulation of Zhang (2008) over PAGs is incomplete. Our approach is complete here and can hence identify the conditional causal effect.

We also compared the running times of Algorithm 1 against

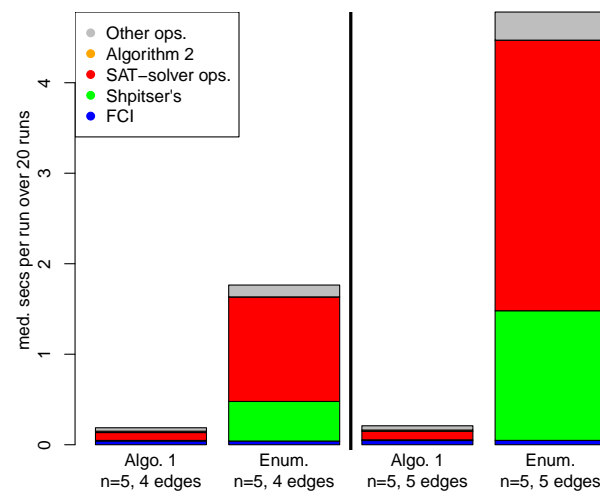


Figure 6: Algorithm 1 vs. Enumeration approach.

trivially enumerating all graphs in the equivalence class and running Shpitser’s algorithm on all of them. Figure 6 shows the median times spent by the different parts of the algorithms. Algorithm 1 is much faster. In addition to the time spent on enumerating the whole equivalence class by the ASP constraint solver used here, running Shpitser’s algorithm on so many graphs also takes a considerable amount of time.

Figure 7 shows the median of the time spent during the different operations of Algorithm 1 on larger instances. FCI was run using an independence oracle. Algorithm 1 spends the majority of its time finding graphs for which the formulas obtained in previous iteration rounds are not warranted. For some outlier instances not visible in the median here, Algorithm 2 also needs a considerable amount of time when finding the derivation for a particularly complicated formula. Note that we undergo here a rather heavy task of finding estimates for all graphs in the equivalence class. If we were content to just decide whether the effect is identifiable, the total running times would be considerably lower. However, as shown in Figure 4, the results of that kind of an algorithm would be quite uninformative.

Finally, we examined the benefits of finding more estimators for the causal effect using Algorithm 2 in cases where multiple different estimators exist. We drew random parameters for a binary SMCM with the graph in Figure 2. Given the equivalence class (shown in Figure 5c), the causal effect  $P(y|do(x = 0))$  can be calculated from the passively observed distribution  $P(h, w, x, y, z)$  either by the backdoor formula adjusting for  $h$ , or by the front-door formula relative to  $z$  (see Figure 5c). We also estimated the causal effect by directly sampling from the model when  $x$  is surgically fixed to 0. Figure 8 shows the average KL-divergence of the different estimators. The distributions

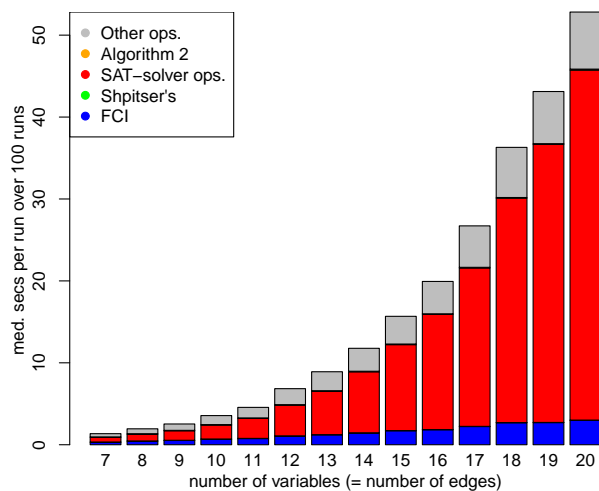


Figure 7: Time spent by Algorithm 1.

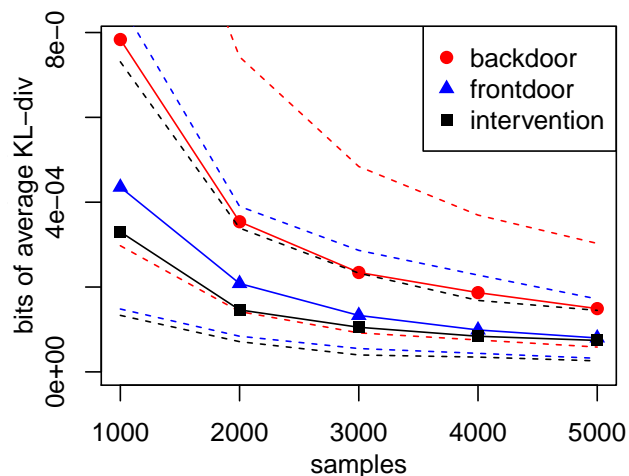


Figure 8: Average KL-divergence for different estimates of  $P(y|do(x))$  for the equivalence class of Figure 2. Median, 33% and 66% quantiles are plotted.

needed for the estimators were estimated directly (with regularization to avoid zero probabilities). In these simulations the front-door estimator seems to offer better accuracy than the backdoor estimator. By intervention we can obtain still higher accuracy than either of the estimates obtained from passively observed data. Shpitser’s algorithm gives here only the backdoor formula. This simulation shows that it may be beneficial to consider many estimators of a causal effect instead of using only a single consistent formula.

## 7 CONCLUSION

In this work we explored the possibilities of estimating causal effects from data. We have considerably relaxed the assumption of the known true graph, which has been standard in the literature on the *do*-calculus. Although causal effects are rarely identified when the true graph is unknown, our approach can still generate informative output in terms of a set of estimates. Unlike other approaches that perform only a limited set of causal effect inferences, our method retains the completeness properties of the used causal discovery algorithm and the *do*-calculus inference. We hope that the flexible machinery presented in this paper can be used to obtain further graphical criteria for identifiability, and will help in achieving more completeness results.

**Acknowledgements** This work was supported in part by the Finnish Foundation for Technology Promotion TES (A.H.), and by the Academy of Finland (grants 251170 COIN Centre of Excellence in Computational Inference Research, 276412 and 284591) and Research Funds of the University of Helsinki (M.J.).

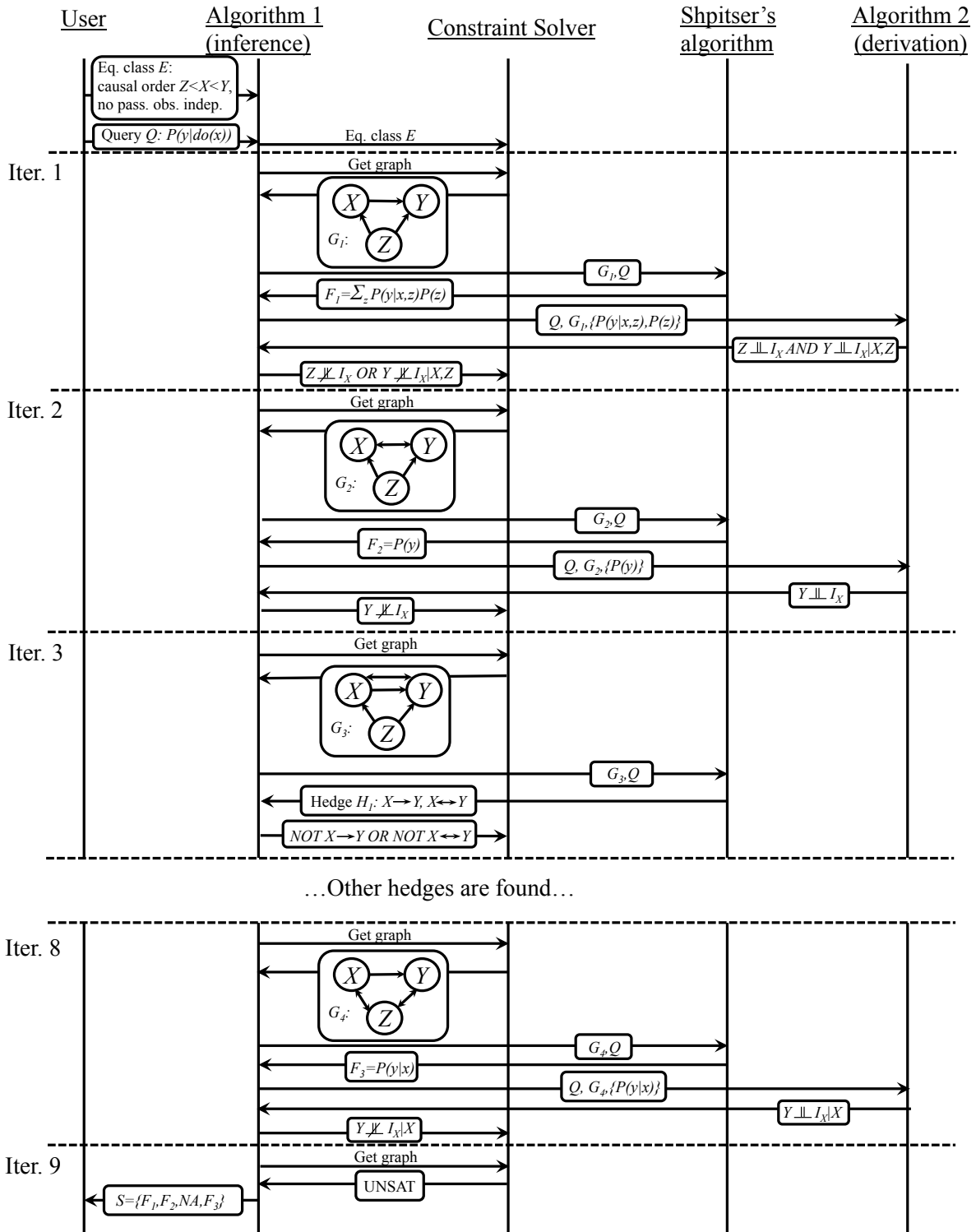


Figure 9: An example run of Algorithm 1 that estimates  $P(y|do(x))$  given the input equivalence class consisting of all SMCM graphs with three variables, no passively observed independencies, and the causal order  $Z < X < Y$ .

## References

- Bareinboim, E. and Pearl, J. (2012). Causal inference by surrogate experiments: z-identifiability. In *Proc. UAI*, pages 113–120. AUAI Press.
- Bareinboim, E. and Pearl, J. (2013). A general algorithm for deciding transportability of experimental results. *Journal of Causal Inference*, 1(1):107–134.
- Biere, A., Heule, M., van Maaren, H., and Walsh, T., editors (2009). *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*. IOS Press.
- Chickering, D. M. (2002). Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3:507–554.
- De Luna, X., Waernbaum, I., and Richardson, T. S. (2011). Covariate selection for the nonparametric estimation of an average treatment effect. *Biometrika*, 98(4):861–875.
- Entner, D., Hoyer, P., and Spirtes, P. (2013). Data-driven covariate selection for nonparametric estimation of causal effects. In *Proc. AISTATS*, volume 31 of *JMLR Workshop and Conference Proceedings*, pages 256–264. JMLR.org.
- Gebser, M., Kaufmann, B., Kaminski, R., Ostrowski, M., Schaub, T., and Schneider, M. T. (2011). Potassco: The Potsdam answer set solving collection. *AI Communications*, 24(2):107–124.
- Hauser, A. and Bühlmann, P. (2012). Characterization and greedy learning of interventional markov equivalence classes of directed acyclic graphs. *Journal of Machine Learning Research*, 13:2409–2464.
- Huang, Y. and Valertorta, M. (2006). Identifiability in causal Bayesian networks: a sound and complete algorithm. In *Proc. AAAI*, pages 1149–1154. AAAI Press.
- Hyttinen, A., Eberhardt, F., and Järvisalo, M. (2014). Constraint-based causal discovery: Conflict resolution with answer set programming. In *Proc. UAI*, pages 340–349. AUAI Press.
- Hyttinen, A., Hoyer, P. O., Eberhardt, F., and Järvisalo, M. (2013). Discovering cyclic causal models with latent variables: A general SAT-based procedure. In *Proc. UAI*, pages 301–310. AUAI Press.
- Kalisch, M., Mächler, M., Colombo, D., Maathuis, M. H., and Bühlmann, P. (2012). Causal inference using graphical models with the R package pcalg. *Journal of Statistical Software*, 47(11):1–26.
- Maathuis, M. H. and Colombo, D. (2015). A generalized backdoor criterion. *Annals of Statistics*, pages 1060–1088.
- Maathuis, M. H., Kalisch, M., Bühlmann, P., et al. (2009). Estimating high-dimensional intervention effects from observational data. *The Annals of Statistics*, 37(6A):3133–3164.
- Malinsky, D. (2015). Estimating intervention effects in systems with unmeasured confounding. Technical report, Carnegie Mellon University.
- Pearl, J. (1995). Causal diagrams for empirical research. *Biometrika*, 82(4):669–688.
- Pearl, J. (2000). *Causality: Models, Reasoning, and Inference*. Cambridge University Press.
- Richardson, T. and Spirtes, P. (2003). Causal inference via ancestral graph models. In Green, P., Hjort, N., and Richardson, S., editors, *Highly Structured Stochastic Systems*, pages 83–105. Oxford University Press.
- Shpitser, I. and Pearl, J. (2006a). Identification of conditional interventional distributions. In *Proc. UAI*, pages 437–444. AUAI Press.
- Shpitser, I. and Pearl, J. (2006b). Identification of joint interventional distributions in recursive semi-markovian causal models. In *Proc. AAAI*, pages 1219–1226. AAAI Press.
- Shpitser, I. and Pearl, J. (2008). Dormant independence. In *Proc. AAAI*, pages 1081–1087. AAAI Press.
- Spirtes, P., Glymour, C., and Scheines, R. (1993). *Causation, Prediction, and Search*. Springer-Verlag. (2nd ed. MIT Press 2000).
- Tian, J. and Pearl, J. (2002). A general identification condition for causal effects. In *Proc. AAAI*, pages 567–573. AAAI Press.
- Tikka, S. (2014). *Package ‘causaleffect’*.
- Tillman, R. E. and Spirtes, P. (2011). Learning equivalence classes of acyclic models with latent and selection variables from multiple datasets with overlapping variables. In *Proc. AISTATS*, volume 15 of *JMLR Proceedings*. JMLR.org.
- Triantafillou, S. and Tsamardinos, I. (2014). Constraint-based causal discovery from multiple interventions over overlapping variable sets. arXiv:1403.2150.
- VanderWeele, T. J. and Shpitser, I. (2011). A new criterion for confounder selection. *Biometrics*, 67(4):1406–1413.
- Zhang, J. (2008). Causal reasoning with ancestral graphs. *The Journal of Machine Learning Research*, 9:1437–1474.

---

# Kernel-Based Just-In-Time Learning for Passing Expectation Propagation Messages

---

Wittawat Jitkrittum,<sup>1</sup> Arthur Gretton,<sup>1</sup> Nicolas Heess,\* S. M. Ali Eslami\*  
Balaji Lakshminarayanan,<sup>1</sup> Dino Sejdinovic<sup>2</sup> and Zoltán Szabó<sup>1</sup>

Gatsby Unit, University College London<sup>1</sup>  
University of Oxford<sup>2</sup>

{wittawatj, arthur.gretton}@gmail.com, nheess@gmail.com  
ali@arkitus.com, balaji@gatsby.ucl.ac.uk,  
dino.sejdinovic@gmail.com, zoltan.szabo@gatsby.ucl.ac.uk

## Abstract

We propose an efficient nonparametric strategy for learning a message operator in expectation propagation (EP), which takes as input the set of incoming messages to a factor node, and produces an outgoing message as output. This learned operator replaces the multivariate integral required in classical EP, which may not have an analytic expression. We use kernel-based regression, which is trained on a set of probability distributions representing the incoming messages, and the associated outgoing messages. The kernel approach has two main advantages: first, it is fast, as it is implemented using a novel two-layer random feature representation of the input message distributions; second, it has principled uncertainty estimates, and can be cheaply updated online, meaning it can request and incorporate new training data when it encounters inputs on which it is uncertain. In experiments, our approach is able to solve learning problems where a single message operator is required for multiple, substantially different data sets (logistic regression for a variety of classification problems), where it is essential to accurately assess uncertainty and to efficiently and robustly update the message operator.

## 1 INTRODUCTION

An increasing priority in Bayesian modelling is to make inference accessible and implementable for practitioners, without requiring specialist knowledge. This is a goal

sought, for instance, in probabilistic programming languages (Wingate et al., 2011; Goodman et al., 2008), as well as in more granular, component-based systems (Stan Development Team, 2014; Minka et al., 2014). In all cases, the user should be able to freely specify what they wish their model to express, without having to deal with the complexities of sampling, variational approximation, or distribution conjugacy. In reality, however, model convenience and simplicity can limit or undermine intended models, sometimes in ways the users might not expect. To take one example, the inverse gamma prior, which is widely used as a convenient conjugate prior for the variance, has quite pathological behaviour (Gelman, 2006). In general, more expressive, freely chosen models are more likely to require expensive sampling or quadrature approaches, which can make them challenging to implement or impractical to run.

We address the particular setting of expectation propagation (Minka, 2001), a message passing algorithm wherein messages are confined to being members of a particular parametric family. The process of integrating incoming messages over a factor potential, and projecting the result onto the required output family, can be difficult, and in some cases not achievable in closed form. Thus, a number of approaches have been proposed to implement EP updates numerically, independent of the details of the factor potential being used. One approach, due to Barthelmé and Chopin (2011), is to compute the message update via importance sampling. While these estimates converge to the desired integrals for a sufficient number of importance samples, the sampling procedure must be run at every iteration during inference, hence it is not viable for large-scale problems.

An improvement on this approach is to use importance sampled instances of input/output message pairs to train a regression algorithm, which can then be used in place

---

\* Currently at Google DeepMind.

of the sampler. Heess et al. (2013) use neural networks to learn the mapping from incoming to outgoing messages, and the learned mappings perform well on a variety of practical problems. This approach comes with a disadvantage: it requires training data that cover the entire set of possible input messages for a given type of problem (e.g., datasets representative of all classification problems the user proposes to solve), and it has no way of assessing the uncertainty of its prediction, or of updating the model online in the event that a prediction is uncertain.

The disadvantages of the neural network approach were the basis for work by Eslami et al. (2014), who replaced the neural networks with random forests. The random forests provide uncertainty estimates for each prediction. This allows them to be trained ‘just-in-time’, during EP inference, whenever the predictor decides it is uncertain. Uncertainty estimation for random forests relies on unproven heuristics, however: we demonstrate empirically that such heuristics can become highly misleading as we move away from the initial training data. Moreover, online updating can result in unbalanced trees, resulting in a cost of prediction of  $O(N)$  for training data of size  $N$ , rather than the ideal of  $O(\log(N))$ .

We propose a novel, kernel-based approach to learning a message operator nonparametrically for expectation propagation. The learning algorithm takes the form of a distribution regression problem, where the inputs are probability measures represented as embeddings of the distributions to a reproducing kernel Hilbert space (RKHS), and the outputs are vectors of message parameters (Szabó et al., 2014). A first advantage of this approach is that one does not need to pre-specify customized features of the distributions, as in (Eslami et al., 2014; Heess et al., 2013). Rather, we use a general characteristic kernel on input distributions (Christmann and Steinwart, 2010, eq. 9). To make the algorithm computationally tractable, we regress directly in the primal from random Fourier features of the data (Rahimi and Recht, 2007; Le et al., 2013; Yang et al., 2015). In particular, we establish a novel random feature representation for when inputs are distributions, via a two-level random feature approach. This gives us both fast prediction (linear in the number of random features), and fast online updates (quadratic in the number of random features).

A second advantage of our approach is that, being an instance of Gaussian process regression, there are well established estimates of predictive uncertainty (Rasmussen and Williams, 2006, Ch. 2). We use these uncertainty estimates so as to determine when to query the importance sampler for additional input/output pairs, i.e., the uncertain predictions trigger just-in-time updates of the regressor. We demonstrate empirically that our uncertainty estimates are more robust and informative than those for random forests, especially as we move away from the training data.

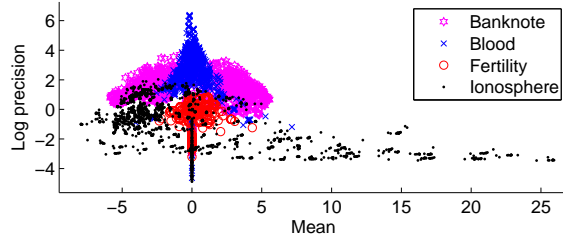


Figure 1: Distributions of incoming messages to logistic factor in four different UCI datasets.

Our paper proceeds as follows. In Section 2, we introduce the notation for expectation propagation, and indicate how an importance sampling procedure can be used as an oracle to provide training data for the message operator. We also give a brief overview of previous learning approaches to the problem, with a focus on that of Eslami et al. (2014). Next, in Section 3, we describe our kernel regression approach, and the form of an efficient kernel message operator mapping the input messages (distributions embedded in an RKHS) to outgoing messages (sets of parameters of the outgoing messages). Finally, in Section 4, we describe our experiments, which cover three topics: a benchmark of our uncertainty estimates, a demonstration of factor learning on artificial data with well-controlled statistical properties, and a logistic regression experiment on four different real-world datasets, demonstrating that our just-in-time learner can correctly evaluate its uncertainty and update the regression function as the incoming messages change (see Fig. 1). Code to implement our method is available online at <https://github.com/wittawatj/kernel-ep>.

## 2 BACKGROUND

We assume that distributions (or densities)  $p$  over a set of variables  $\mathbf{x} = (x_1, \dots, x_d)$  of interest can be represented as factor graphs, i.e.  $p(\mathbf{x}) = \frac{1}{Z} \prod_{j=1}^J f_j(\mathbf{x}_{\text{ne}(f_j)})$ . The factors  $f_j$  are non-negative functions which are defined over subsets  $\mathbf{x}_{\text{ne}(f_j)}$  of the full set of variables  $\mathbf{x}$ . These variables form the neighbors of the factor node  $f_j$  in the factor graph, and we use  $\text{ne}(f_j)$  to denote the corresponding set of indices.  $Z$  is the normalization constant.

We deal with models in which some of the factors have a non-standard form, or may not have a known analytic expression (i.e. ‘‘black box’’ factors). Although our approach applies to any such factor in principle, in this paper we focus on *directed* factors  $f(\mathbf{x}_{\text{out}}|\mathbf{x}_{\text{in}})$  which specify a conditional distribution over variables  $\mathbf{x}_{\text{out}}$  given  $\mathbf{x}_{\text{in}}$  (and thus  $\mathbf{x}_{\text{ne}(f)} = (\mathbf{x}_{\text{out}}, \mathbf{x}_{\text{in}})$ ). The only assumption we make is that we are provided with a forward sampling function  $f : \mathbf{x}_{\text{in}} \mapsto \mathbf{x}_{\text{out}}$ , i.e., a function that maps (stochastically or deterministically) a setting of the input variables  $\mathbf{x}_{\text{in}}$  to a sample from the conditional distribution



over  $\mathbf{x}_{\text{out}} \sim f(\cdot|\mathbf{x}_{\text{in}})$ . In particular, the ability to evaluate the value of  $f(\mathbf{x}_{\text{out}}|\mathbf{x}_{\text{in}})$  is not assumed. A natural way to specify  $f$  is as code in a probabilistic program.

## 2.1 EXPECTATION PROPAGATION

Expectation Propagation (EP) is an approximate iterative procedure for computing marginal beliefs of variables by iteratively passing messages between variables and factors until convergence (Minka, 2001). It can be seen as an alternative to belief propagation, where the marginals are projected onto a member of some class of known parametric distributions. The message  $m_{f \rightarrow V}(x_V)$  from factor  $f$  to variable  $V \in \text{ne}(f)$  is

$$\frac{\text{proj} \left[ \int f(\mathbf{x}_{\text{ne}(f)}) \prod_{V' \in \text{ne}(f)} m_{V' \rightarrow f}(x_{V'}) d\mathbf{x}_{\text{ne}(f) \setminus V} \right]}{m_{V \rightarrow f}(x_V)}, \quad (1)$$

where  $m_{V' \rightarrow f}$  are the messages sent to factor  $f$  from all of its neighboring variables  $x_{V'}$ ,  $\text{proj}[p] = \text{argmin}_{q \in \mathcal{Q}} \text{KL}[p||q]$ , and  $\mathcal{Q}$  is typically in the exponential family, e.g. the set of Gaussian or Beta distributions.

Computing the numerator of (1) can be challenging, as it requires evaluating a high-dimensional integral as well as minimization of the Kullback-Leibler divergence to some non-standard distribution. Even for factors with known analytic form this often requires hand-crafted approximations, or the use of expensive numerical integration techniques; for “black-box” factors implemented as forward sampling functions, fully nonparametric techniques are needed.

Barthelmé and Chopin (2011); Heess et al. (2013); Eslami et al. (2014) propose an alternative, stochastic approach to the integration and projection step. When the projection is to a member  $q(x|\eta) = h(x) \exp(\eta^\top u(x) - A(\eta))$  of an exponential family, one simply computes the expectation of the sufficient statistic  $u(\cdot)$  under the numerator of (1). A sample based approximation of this expectation can be obtained via Monte Carlo simulation. Given a forward-sampling function  $f$  as described above, one especially simple approach is importance sampling,

$$\mathbb{E}_{\mathbf{x}_{\text{ne}(f)} \sim b} [u(x_V)] \approx \frac{1}{M} \sum_{l=1}^M w(\mathbf{x}_{\text{ne}(f)}^l) u(x_V^l), \quad (2)$$

where  $\mathbf{x}_{\text{ne}(f)}^l \sim \tilde{b}$ , for  $l = 1, \dots, M$  and on the left hand side,

$$b(\mathbf{x}_{\text{ne}(f)}) = f(\mathbf{x}_{\text{ne}(f)}) \prod_{W \in \text{ne}(f)} m_{W \rightarrow f}(x_W).$$

On the right hand side we draw samples  $\mathbf{x}_{\text{ne}(f)}^l$  from some proposal distribution  $\tilde{b}$  which we choose to be  $\tilde{b}(\mathbf{x}_{\text{ne}(f)}) = r(\mathbf{x}_{\text{in}})f(\mathbf{x}_{\text{out}}|\mathbf{x}_{\text{in}})$  for some distribution  $r$  with appropriate

support, and compute importance weights

$$w(\mathbf{x}_{\text{ne}(f)}) = \frac{\prod_{W \in \text{ne}(f)} m_{W \rightarrow f}(x_W)}{r(\mathbf{x}_{\text{in}})}.$$

Thus the estimated expected sufficient statistics provide us with an estimate of the parameters  $\eta$  of the result  $q$  of the projection  $\text{proj}[p]$ , from which the message is readily computed.

## 2.2 JUST-IN-TIME LEARNING OF MESSAGES

Message approximations as in the previous section could be used directly when running the EP algorithm, as in Barthelmé and Chopin (2011), but this approach can suffer when the number of samples  $M$  is small, and the importance sampling estimate is not reliable. On the other hand, for large  $M$  the computational cost of running EP with approximate messages can be very high, as importance sampling must be performed for sending each outgoing message. To obtain low-variance message approximations at lower computational cost, Heess et al. (2013) and Eslami et al. (2014) both amortize previously computed approximate messages by training a function approximator to directly map a tuple of incoming variable-to-factor messages  $(m_{V' \rightarrow f})_{V' \in \text{ne}(f)}$  to an approximate factor to variable message  $m_{f \rightarrow V}$ , i.e. they learn a mapping

$$M_{f \rightarrow V}^\theta : (m_{V' \rightarrow f})_{V' \in \text{ne}(f)} \mapsto m_{f \rightarrow V}, \quad (3)$$

where  $\theta$  are the parameters of the approximator.

Heess et al. (2013) use neural networks and a large, fixed training set to learn their approximate message operator prior to running EP. By contrast, Eslami et al. (2014) employ random forests as their class of learning functions, and update their approximate message operator on the fly during inference, depending on the predictive uncertainty of the current message operator. Specifically, they endow their function approximator with an uncertainty estimate

$$\mathfrak{V}_{f \rightarrow V}^\theta : (m_{V' \rightarrow f})_{V' \in \text{ne}(f)} \mapsto \mathbf{v}, \quad (4)$$

where  $\mathbf{v}$  indicates the expected unreliability of the predicted, approximate message  $m_{f \rightarrow V}$  returned by  $M_{f \rightarrow V}^\theta$ . If  $\mathbf{v} = \mathfrak{V}_{f \rightarrow V}^\theta((m_{V' \rightarrow f})_{V' \in \text{ne}(f)})$  exceeds a predefined threshold, the required message is approximated via importance sampling (cf. (2)) and  $M_{f \rightarrow V}^\theta$  is updated on this new datapoint (leading to a new set of parameters  $\theta'$  with  $\mathfrak{V}_{f \rightarrow V}^{\theta'}((m_{V' \rightarrow f})_{V' \in \text{ne}(f)}) < \mathfrak{V}_{f \rightarrow V}^\theta((m_{V' \rightarrow f})_{V' \in \text{ne}(f)})$ ).

Eslami et al. (2014) estimate the predictive uncertainty  $\mathfrak{V}_{f \rightarrow V}^\theta$  via the heuristic of looking at the variability of the forest predictions for each point (Criminisi and Shotton, 2013). They implement their online updates by splitting the trees at their leaves. Both these mechanisms can be problematic, however. First, the heuristic used in computing uncertainty has no guarantees: indeed, uncertainty estimation

for random forests remains a challenging topic of current research (Hutter, 2009). This is not merely a theoretical consideration: in our experiments in Section 4, we demonstrate that uncertainty heuristics for random forests become unstable and inaccurate as we move away from the initial training data. Second, online updates of random forests may not work well when the newly observed data are from a very different distribution to the initial training sample (e.g. Lakshminarayanan et al., 2014, Fig. 3). For large amounts of training set drift, the leaf-splitting approach of Eslami et al. can result in a decision tree in the form of a long chain, giving a worst case cost of prediction (computational and storage) of  $O(N)$  for training data of size  $N$ , vs the ideal of  $O(\log(N))$  for balanced trees. Finally, note that the approach of Eslami et al. uses certain bespoke features of the factors when specifying tree traversal in the random forests, notably the value of the factor potentials at the mean and mode of the incoming messages. These features require expert knowledge of the model on the part of the practitioner, and are not available in the “forward sampling” setting. The present work does not employ such features.

In terms of computational cost, prediction for the random forest of Eslami et al. costs  $O(KD_r D_t \log(N))$ , and updating following a new observation costs  $O(KD_r^3 D_t \log(N))$ , where  $K$  is the number of trees in the random forest,  $D_t$  is the number of features used in tree traversal,  $D_r$  is the number of features used in making predictions at the leaves, and  $N$  is the number of training messages. Representative values are  $K = 64$ ,  $D_t = D_r \approx 15$ , and  $N$  in the range of 1,000 to 5,000.

### 3 KERNEL LEARNING OF OPERATORS

We now propose a kernel regression method for jointly learning the message operator  $M_{f \rightarrow V}^\theta$  and uncertainty estimate  $\mathfrak{A}_{f \rightarrow V}^\theta$ . We regress from the tuple of incoming messages, which are probability distributions, to the parameters of the outgoing message. To this end we apply a kernel over distributions from (Christmann and Steinwart, 2010) to the case where the input consists of more than one distribution.

We note that Song et al. (2010, 2011) propose a related regression approach for predicting outgoing messages from incoming messages, for the purpose of belief propagation. Their setting is different from ours, however, as their messages are smoothed conditional density functions rather than parametric distributions of known form.

To achieve fast predictions and factor updates, we follow Rahimi and Recht (2007); Le et al. (2013); Yang et al. (2015), and express the kernel regression in terms of random features whose expected inner product is equal to the kernel function; i.e. we perform regression directly in the primal on these random features. In Section 3.1, we de-

fine our kernel on tuples of distributions, and then derive the corresponding random feature representation in Section 3.2. Section 3.3 describes the regression algorithm, as well as our strategy for uncertainty evaluation and online updates.

#### 3.1 KERNELS ON TUPLES OF DISTRIBUTIONS

In the following, we consider only a single factor, and therefore drop the factor identity from our notation. We write the set of  $c$  incoming messages to a factor node as a tuple of probability distributions  $R := (r^{(l)})_{l=1}^c$  of random variables  $X^{(l)}$  on respective domains  $\mathcal{X}^{(l)}$ . Our goal is to define a kernel between one such tuple, and a second one, which we will write  $S := (s^{(l)})_{l=1}^c$ .

We define our kernel in terms of embeddings of the tuples  $R, S$  into a reproducing kernel Hilbert space (RKHS). We first consider the embedding of a single distribution in the tuple: Let us define an RKHS  $\mathcal{H}^{(l)}$  on each domain, with respective kernel  $k^{(l)}(x_1^{(l)}, x_2^{(l)})$ . We may embed individual probability distributions to these RKHSs, following (Smola et al., 2007). The *mean embedding* of  $r^{(l)}$  is written

$$\mu_{r^{(l)}}(\cdot) := \int k^{(l)}(x^{(l)}, \cdot) dr^{(l)}(x^{(l)}). \quad (5)$$

Similarly, a mean embedding may be defined on the product of messages in a tuple  $r = \times_{l=1}^c r^{(l)}$  as

$$\mu_r := \int k([x^{(1)}, \dots, x^{(c)}], \cdot) dr(x^{(1)}, \dots, x^{(c)}), \quad (6)$$

where we have defined the joint kernel  $k$  on the product space  $\mathcal{X}^{(1)} \times \dots \times \mathcal{X}^{(c)}$ . Finally, a kernel on two such embeddings  $\mu_r, \mu_s$  of tuples  $R, S$  can be obtained as in Christmann and Steinwart (2010, eq. 9),

$$\kappa(r, s) = \exp\left(-\frac{\|\mu_r - \mu_s\|_{\mathcal{H}}^2}{2\gamma^2}\right). \quad (7)$$

This kernel has two parameters:  $\gamma^2$ , and the width parameter of the kernel  $k$  defining  $\mu_r = \mathbb{E}_{x \sim r} k(x, \cdot)$ .

We have considered several alternative kernels on tuples of messages, including kernels on the message parameters, kernels on a tensor feature space of the distribution embeddings in the tuple, and dot products of the features (6). We have found these alternatives to have worse empirical performance than the approach described above. We give details of these experiments in Section C of the supplementary material.

#### 3.2 RANDOM FEATURE APPROXIMATIONS

One approach to learning the mapping  $M_{f \rightarrow V}^\theta$  from incoming to outgoing messages would be to employ Gaussian process regression, using the kernel (7). This approach is

not suited to just-in-time (JIT) learning, however, as both prediction and storage costs grow with the size of the training set; thus, inference on even moderately sized datasets rapidly becomes computationally prohibitive. Instead, we define a finite-dimensional random feature map  $\hat{\psi} \in \mathbb{R}^{D_{\text{out}}}$  such that  $\kappa(r, s) \approx \hat{\psi}(r)^\top \hat{\psi}(s)$ , and regress directly on these feature maps in the primal (see next section): storage and computation are then a function of the dimension of the feature map  $D_{\text{out}}$ , yet performance is close to that obtained using a kernel.

In [Rahimi and Recht \(2007\)](#), a method based on Fourier transforms was proposed for computing a vector of random features  $\hat{\varphi}$  for a translation invariant kernel  $k(x, y) = k(x - y)$  such that  $k(x, y) \approx \hat{\varphi}(x)^\top \hat{\varphi}(y)$  where  $x, y \in \mathbb{R}^d$  and  $\hat{\varphi}(x), \hat{\varphi}(y) \in \mathbb{R}^{D_{\text{in}}}$ . This is possible because of Bochner’s theorem ([Rudin, 2013](#)), which states that a continuous, translation-invariant kernel  $k$  can be written in the form of an inverse Fourier transform:

$$k(x - y) = \int \hat{k}(\omega) e^{j\omega^\top (x-y)} d\omega,$$

where  $j = \sqrt{-1}$  and the Fourier transform  $\hat{k}$  of the kernel can be treated as a distribution. The inverse Fourier transform can thus be seen as an expectation of the complex exponential, which can be approximated with a Monte Carlo average by drawing random frequencies from the Fourier transform. We will follow a similar approach, and derive a two-stage set of random Fourier features for (7).

We start by expanding the exponent of (7) as

$$\exp\left(-\frac{1}{2\gamma^2} \langle \mu_r, \mu_r \rangle + \frac{1}{\gamma^2} \langle \mu_r, \mu_s \rangle - \frac{1}{2\gamma^2} \langle \mu_s, \mu_s \rangle\right).$$

Assume that the embedding kernel  $k$  used to define the embeddings  $\mu_r$  and  $\mu_s$  is translation invariant. Since  $\langle \mu_r, \mu_s \rangle = \mathbb{E}_{x \sim r} \mathbb{E}_{y \sim s} k(x - y)$ , one can use the result of [Rahimi and Recht \(2007\)](#) to write

$$\begin{aligned} \langle \mu_r, \mu_s \rangle &\approx \mathbb{E}_{x \sim r} \mathbb{E}_{y \sim s} \hat{\varphi}(x)^\top \hat{\varphi}(y) \\ &= \mathbb{E}_{x \sim r} \hat{\varphi}(x)^\top \mathbb{E}_{y \sim s} \hat{\varphi}(y) := \hat{\varphi}(r)^\top \hat{\varphi}(s), \end{aligned}$$

where the mappings  $\hat{\varphi}$  are  $D_{\text{in}}$  standard Rahimi-Recht random features, shown in Steps 1-3 of Algorithm 1.

With the approximation of  $\langle \mu_r, \mu_s \rangle$ , we have

$$\kappa(r, s) \approx \exp\left(-\frac{\|\hat{\varphi}(r) - \hat{\varphi}(s)\|_{D_{\text{in}}}^2}{2\gamma^2}\right), \quad (8)$$

which is a standard Gaussian kernel on  $\mathbb{R}^{D_{\text{in}}}$ . We can thus further approximate this Gaussian kernel by the random Fourier features of [Rahimi and Recht](#), to obtain a vector of random features  $\hat{\psi}$  such that  $\kappa(r, s) \approx \hat{\psi}(r)^\top \hat{\psi}(s)$  where  $\hat{\psi}(r), \hat{\psi}(s) \in \mathbb{R}^{D_{\text{out}}}$ . Pseudocode for generating the random features  $\hat{\psi}$  is given in Algorithm 1. Note that the sine

---

**Algorithm 1** Construction of two-stage random features for  $\kappa$

---

**Input:** Input distribution  $r$ , Fourier transform  $\hat{k}$  of the embedding translation-invariant kernel  $k$ , number of inner features  $D_{\text{in}}$ , number of outer features  $D_{\text{out}}$ , outer Gaussian width  $\gamma^2$ .

**Output:** Random features  $\hat{\psi}(r) \in \mathbb{R}^{D_{\text{out}}}$ .

- 1: Sample  $\{\omega_i\}_{i=1}^{D_{\text{in}}} \stackrel{i.i.d.}{\sim} \hat{k}$ .
- 2: Sample  $\{b_i\}_{i=1}^{D_{\text{in}}} \stackrel{i.i.d.}{\sim} \text{Uniform}[0, 2\pi]$ .
- 3:  $\hat{\phi}(r) = \sqrt{\frac{2}{D_{\text{in}}}} \left( \mathbb{E}_{x \sim r} \cos(\omega_i^\top x + b_i) \right)_{i=1}^{D_{\text{in}}} \in \mathbb{R}^{D_{\text{in}}}$   
If  $r(x) = \mathcal{N}(x; m, \Sigma)$ ,

$$\hat{\phi}(r) = \sqrt{\frac{2}{D_{\text{in}}}} \left( \cos(\omega_i^\top m + b_i) \exp\left(-\frac{1}{2} \omega_i^\top \Sigma \omega_i\right) \right)_{i=1}^{D_{\text{in}}}.$$

- 4: Sample  $\{\nu_i\}_{i=1}^{D_{\text{out}}} \stackrel{i.i.d.}{\sim} \hat{k}_{\text{gauss}}(\gamma^2)$  i.e., Fourier transform of a Gaussian kernel with width  $\gamma^2$ .
  - 5: Sample  $\{c_i\}_{i=1}^{D_{\text{out}}} \stackrel{i.i.d.}{\sim} \text{Uniform}[0, 2\pi]$ .
  - 6:  $\hat{\psi}(r) = \sqrt{\frac{2}{D_{\text{out}}}} \left( \cos(\nu_i^\top \hat{\phi}(r) + c_i) \right)_{i=1}^{D_{\text{out}}} \in \mathbb{R}^{D_{\text{out}}}$
- 

component in the complex exponential vanishes due to the translation invariance property (analogous to an even function), i.e., only the cosine term remains. We refer to Section B.3 in the supplementary material for more details.

For the implementation, we need to pre-compute  $\{\omega_i\}_{i=1}^{D_{\text{in}}}, \{b_i\}_{i=1}^{D_{\text{in}}}, \{\nu_i\}_{i=1}^{D_{\text{out}}}$  and  $\{c_i\}_{i=1}^{D_{\text{out}}}$ , where  $D_{\text{in}}$  and  $D_{\text{out}}$  are the number of random features used. A more efficient way to support a large number of random features is to store only the random seed used to generate the features, and to generate the coefficients on-the-fly as needed ([Dai et al., 2014](#)). In our implementation, we use a Gaussian kernel for  $k$ .

### 3.3 REGRESSION FOR OPERATOR PREDICTION

Let  $\mathbf{X} = (x_1 | \dots | x_N)$  be the  $N$  training samples of incoming messages to a factor node, and let  $\mathbf{Y} = \left( \mathbb{E}_{x_V \sim q_{f \rightarrow V}^1} u(x_V) | \dots | \mathbb{E}_{x_V \sim q_{f \rightarrow V}^N} u(x_V) \right) \in \mathbb{R}^{D_y \times N}$  be the expected sufficient statistics of the corresponding output messages, where  $q_{f \rightarrow V}^i$  is the numerator of (1). We write  $x_i = \hat{\psi}(r_i)$  as a more compact notation for the random feature vector representing the  $i^{\text{th}}$  training tuple of incoming messages, as computed via Algorithm 1.

Since we require uncertainty estimates on our predictions, we perform Bayesian linear regression from the random features to the output messages, which yields predictions close to those obtained by Gaussian process regression with the kernel in (7). The uncertainty estimate in this case will

be the predictive variance. We assume prior and likelihood

$$w \sim \mathcal{N}(w; 0, I_{D_{\text{out}}}\sigma_0^2), \quad (9)$$

$$Y | X, w \sim \mathcal{N}(Y; w^\top X, \sigma_y^2 I_N), \quad (10)$$

where the output noise variance  $\sigma_y^2$  captures the intrinsic stochasticity of the importance sampler used to generate  $Y$ . It follows that the posterior of  $w$  is given by (Bishop, 2006)

$$p(w|Y) = \mathcal{N}(w; \mu_w, \Sigma_w), \quad (11)$$

$$\Sigma_w = (\mathbf{X}\mathbf{X}^\top \sigma_y^{-2} + \sigma_0^{-2}I)^{-1}, \quad (12)$$

$$\mu_w = \Sigma_w \mathbf{X}Y^\top \sigma_y^{-2}. \quad (13)$$

The predictive distribution on the output  $y^*$  given an observation  $x^*$  is

$$p(y^*|x^*, Y) = \int p(y^*|w, x^*, Y)p(w|Y) dw \quad (14)$$

$$= \mathcal{N}(y^*; x^{*\top} \mu_w, x^{*\top} \Sigma_w x^* + \sigma_y^2). \quad (15)$$

For simplicity, we treat each output (expected sufficient statistic) as a separate regression problem. Treating all outputs jointly can be achieved with a multi-output kernel (Alvarez et al., 2011).

**Online Update** We describe an online update for  $\Sigma_w$  and  $\mu_w$  when observations (i.e., random features representing incoming messages)  $x_i$  arrive sequentially. We use  $\cdot^{(N)}$  to denote a quantity constructed from  $N$  samples. Recall that  $\Sigma_w^{-1(N)} = \mathbf{X}\mathbf{X}^\top \sigma_y^{-2} + \sigma_0^{-2}I$ . The posterior covariance matrix at time  $N + 1$  is

$$\Sigma_w^{(N+1)} = \Sigma_w^{(N)} - \frac{\Sigma_w^{(N)} x_{N+1} x_{N+1}^\top \Sigma_w^{(N)} \sigma_y^{-2}}{1 + x_{N+1}^\top \Sigma_w^{(N)} x_{N+1} \sigma_y^{-2}}, \quad (16)$$

meaning it can be expressed as an inexpensive update of the covariance at time  $N$ . Updating  $\Sigma_w$  for all the  $D_y$  outputs costs  $O((D_{\text{in}}D_{\text{out}} + D_{\text{out}}^2)D_y)$  per new observation. For  $\mu_w = \Sigma_w \mathbf{X}Y^\top \sigma_y^{-2}$ , we maintain  $\mathbf{X}Y^\top \in \mathbb{R}^{D_{\text{out}} \times D_y}$ , and update it at cost  $O(D_{\text{in}}D_{\text{out}}D_y)$  as

$$(\mathbf{X}Y^\top)^{(N+1)} = (\mathbf{X}Y^\top + x_{N+1}y_{N+1}^\top). \quad (17)$$

Since we have  $D_y$  regression functions, for each tuple of incoming messages  $x^*$ , there are  $D_y$  predictive variances,  $v_1^*, \dots, v_{D_y}^*$ , one for each output. Let  $\{\tau_i\}_{i=1}^{D_y}$  be pre-specified predictive variance thresholds. Given a new input  $x^*$ , if  $v_1^* > \tau_1$  or  $\dots$  or  $v_{D_y}^* > \tau_{D_y}$  (the operator is uncertain), a query is made to the oracle to obtain a ground truth  $y^*$ . The pair  $(x^*, y^*)$  is then used to update  $\Sigma_w$  and  $\mu_w$ .

## 4 EXPERIMENTS

We evaluate our learned message operator using two different factors: the logistic factor, and the compound gamma

factor. In the first and second experiment we demonstrate that the proposed operator is capable of learning high-quality mappings from incoming to outgoing messages, and that the associated uncertainty estimates are reliable. The third and fourth experiments assess the performance of the operator as part of the full EP inference loop in two different models: approximating the logistic, and the compound gamma factor. Our final experiment demonstrates the ability of our learning process to reliably and quickly adapt to large shifts in the message distribution, as encountered during inference in a sequence of several real-world regression problems.

For all experiments we used Infer.NET (Minka et al., 2014) with its extensible factor interface for our own operator. We used the default settings of Infer.NET unless stated otherwise. The regression target is the marginal belief (numerator of (1)) in experiment 1,2,3 and 5. We set the regression target to the outgoing message in experiment 4. Given a marginal belief, the outgoing message can be calculated straightforwardly.

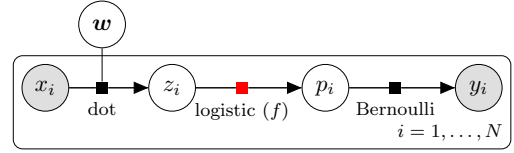


Figure 2: Factor graph for binary logistic regression. The kernel-based message operator learns to approximate the logistic factor highlighted in red. The two incoming messages are  $m_{z_i \rightarrow f} = \mathcal{N}(z_i; \mu, \sigma^2)$  and  $m_{p_i \rightarrow f} = \text{Beta}(p_i; \alpha, \beta)$ .

**Experiment 1: Batch Learning** As in (Heess et al., 2013; Eslami et al., 2014), we study the logistic factor  $f(p|z) = \delta\left(p - \frac{1}{1+\exp(-z)}\right)$ , where  $\delta$  is the Dirac delta function, in the context of a binary logistic regression model (Fig. 2). The factor is deterministic and there are two incoming messages:  $m_{p_i \rightarrow f} = \text{Beta}(p_i; \alpha, \beta)$  and  $m_{z_i \rightarrow f} = \mathcal{N}(z_i; \mu, \sigma^2)$ , where  $z_i = w^\top x_i$  represents the dot product between an observation  $x_i \in \mathbb{R}^d$  and the coefficient vector  $w$  whose posterior is to be inferred.

In this first experiment we simply learn a kernel-based operator to send the message  $m_{f \rightarrow z_i}$ . Following Eslami et al. (2014), we set  $d$  to 20, and generated 20 different datasets, sampling a different  $w \sim \mathcal{N}(0, I)$  and then a set of  $\{(x_i, y_i)\}_{i=1}^n$  ( $n = 300$ ) observations according to the model. For each dataset we ran EP for 10 iterations, and collected incoming-outgoing message pairs in the first five iterations of EP from Infer.NET’s implementation of the logistic factor. We partitioned the messages randomly into 5,000 training and 3,000 test messages, and learned a message operator to predict  $m_{f \rightarrow z_i}$  as described in Section 3. Regularization and kernel parameters were chosen

by leave-one-out cross validation. We set the number of random features to  $D_{in} = 500$  and  $D_{out} = 1,000$ ; empirically, we observed no significant improvements beyond 1,000 random features.

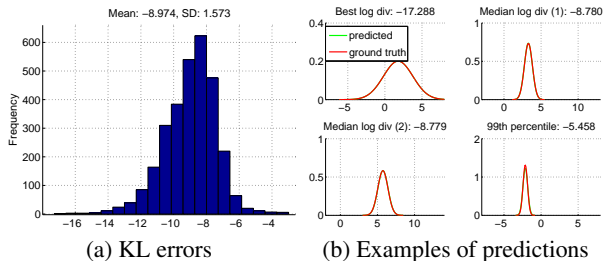


Figure 3: Prediction errors for predicting the projected beliefs to  $z_i$ , and examples of predicted messages at different error levels.

We report  $\log \text{KL}[q_{f \rightarrow z_i} || \hat{q}_{f \rightarrow z_i}]$  where  $q_{f \rightarrow z_i}$  is the ground truth projected belief (numerator of (1)) and  $\hat{q}_{f \rightarrow z_i}$  is the prediction. The histogram of the log KL errors is shown in Fig. 3a; Fig. 3b shows examples of predicted messages for different log KL errors. It is evident that the kernel-based operator does well in capturing the relationship between incoming and outgoing messages. The discrepancy with respect to the ground truth is barely visible even at the 99th percentile. See Section C in the supplementary material for a comparison with other methods.

**Experiment 2: Uncertainty Estimates** For the approximate message operator to perform well in a JIT learning setting, it is crucial to have reliable estimates of operator’s predictive uncertainty in different parts of the space of incoming messages. To assess this property we compute the predictive variance using the same learned operator as used in Fig. 3. The forward incoming messages  $m_{z_i \rightarrow f}$  in the previously used training set are shown in Fig. 4a. The backward incoming messages  $m_{p_i \rightarrow f}$  are not displayed. Shown in the same plot are two curves (a blue line, and a pink parabola) representing two “uncertainty test sets”: these are the sets of parameter pairs on which we wish to evaluate the uncertainty of the predictor, and pass through regions with both high and low densities of training samples. Fig. 4b shows uncertainty estimates of our kernel-based operator and of random forests, where we fix  $m_{p_i \rightarrow f} := \text{Beta}(p_i; 1, 2)$  for testing. The implementation of the random forests closely follows Eslami et al. (2014).

From the figure, as the mean of the test message moves away from the region densely sampled by the training data, the predictive variance reported by the kernel method increases much more smoothly than that of the random forests. Further, our method clearly exhibits a higher uncertainty on the test set #1 than on the test set #2. This

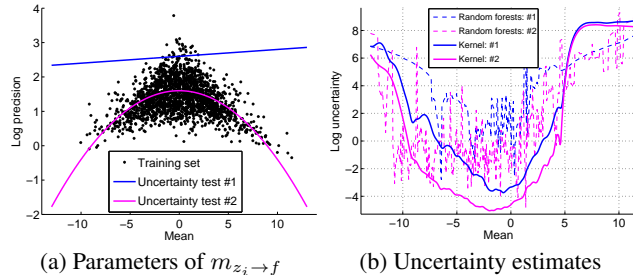


Figure 4: (a) Incoming messages from  $z$  to  $f$  from 20 EP runs of binary logistic regression, as shown in Fig. 2. (b) Uncertainty estimates of the proposed kernel-based method (predictive variance) and Eslami et al.’s random forests (KL-based agreement of predictions of different trees) on the two uncertainty test sets shown. For testing, we fix the other incoming message  $m_{p_i \rightarrow f}$  to  $\text{Beta}(p_i; 1, 2)$ .

behaviour is desirable, as most of the points in test set #1 are either in a low density region or an unexplored region. These results suggest that the predictive variance is a robust criterion for querying the importance sampling oracle. One key observation is that the uncertainty estimates of the random forests are highly non-smooth; i.e., uncertainty on nearby points may vary wildly. As a result, a random forest-based JIT learner may still query the importance sampler oracle when presented with incoming messages similar to those in the training set, thereby wasting computation.

We have further checked that the predictive uncertainty of the regression function is a reliable indication of the error in KL divergence of the predicted outgoing messages. These results are given in Figure 10 of Appendix C.

**Experiment 3: Just-In-Time Learning** In this experiment we test the approximate operator in the logistic regression model as part of the full EP inference loop in a just-in-time learning setting (KJIT). We now learn two kernel-based message operators, one for each outgoing direction from the logistic factor. The data generation is the same as in the batch learning experiment. We sequentially presented the operator with 30 related problems, where a new set of observations  $\{(x_i, y_i)\}_{i=1}^n$  was generated at the beginning of each problem from the model, while keeping  $w$  fixed. This scenario is common in practice: one is often given several sets of observations which share the same model parameter (Eslami et al., 2014). As before, the inference target was  $p(w | \{(x_i, y_i)\}_{i=1}^n)$ . We set the maximum number of EP iterations to 10 in each problem.

We employed a “mini-batch” learning approach in which the operator always consults the oracle in the first few hundred factor invocations for initial batch training. In principle, during the initial batch training, the operator can perform cross validation or type-II maximum likelihood esti-

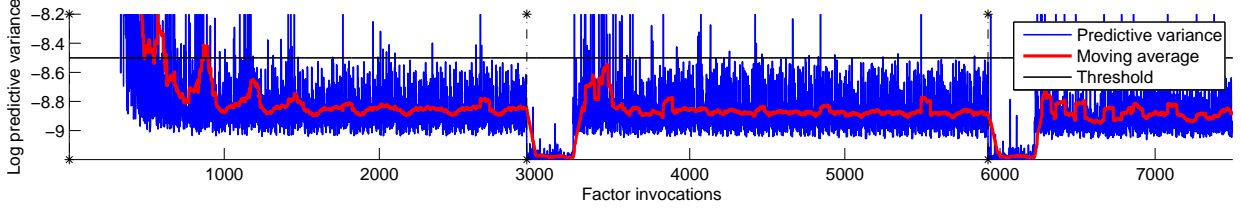


Figure 5: Uncertainty estimate of KJIT in its prediction of outgoing messages at each factor invocation, for the binary regression problem. The black dashed lines indicate the start of a new inference problem.

mation for parameter selection; however for computational simplicity we set the kernel parameters according to the median heuristic (Schölkopf and Smola, 2002). Full detail of the heuristic is given in Section A in the supplementary material. The numbers of random features were  $D_{\text{in}} = 300$  and  $D_{\text{out}} = 500$ . The output noise variance  $\sigma_y^2$  was fixed to  $10^{-4}$  and the uncertainty threshold on the log predictive variance was set to  $-8.5$ . To simulate a black-box setup, we used an importance sampler as the oracle rather than Infer.NET’s factor implementation, where the proposal distribution was fixed to  $\mathcal{N}(z; 0, 200)$  with  $5 \times 10^5$  particles.

Fig. 5 shows a trace of the predictive variance of KJIT in predicting the mean of each  $m_{f \rightarrow z_i}$  upon each factor invocation. The black dashed lines indicate the start of a new inference problem. Since the first 300 factor invocations are for the initial training, no uncertainty estimate is shown. From the trace, we observe that the uncertainty rapidly drops down to a stable point at roughly  $-8.8$  and levels off after the operator sees about 1,000 incoming-outgoing message pairs, which is relatively low compared to approximately 3,000 message passings (i.e., 10 iterations  $\times$  300 observations) required for one problem. The uncertainty trace displays a periodic structure, repeating itself in every 300 factor invocations, corresponding to a full sweep over all 300 observations to collect incoming messages  $m_{z_i \rightarrow f}$ . The abrupt drop in uncertainty in the first EP iteration of each new problem is due to the fact that Infer.NET’s inference engine initializes the message from  $w$  to have zero mean, leading to  $m_{z_i \rightarrow f}$  also having a zero mean. Repeated encounters of such a zero mean incoming message reinforce the operator’s confidence; hence the drop in uncertainty.

Fig. 6a shows binary classification errors obtained by using the inferred posterior mean of  $w$  on a test set of size 10000 generated from the true underlying parameter. Included in the plot are the errors obtained by using only the importance sampler for inference (“Sampling”), and using the Infer.NET’s hand-crafted logistic factor. The loss of KJIT matches well with that of the importance sampler and Infer.NET, suggesting that the inference accuracy is as good as these alternatives. Fig. 6b shows the inference time required by all methods in each problem. While the inference quality is equally good, KJIT is orders of magnitude faster

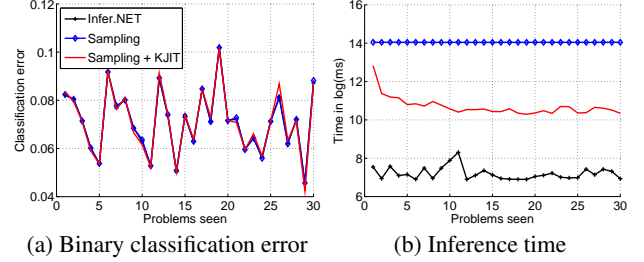


Figure 6: Classification performance and inference times of all methods in the binary logistic regression problem.

than the importance sampler.

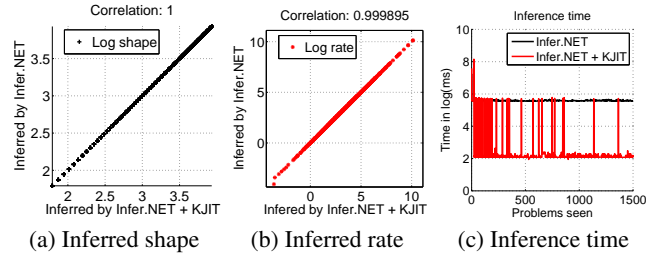


Figure 7: Shape (a) and rate (b) parameters of the inferred posteriors in the compound gamma problem. (c) KJIT is able to infer equally good posterior parameters compared to Infer.NET, while requiring a runtime several orders of magnitude lower.

**Experiment 4: Compound Gamma Factor** We next simulate the compound gamma factor, a heavy-tailed prior distribution on the precision of a Gaussian random variable. A variable  $\tau$  is said to follow the compound gamma distribution if  $\tau \sim \text{Gamma}(\tau; s_2, r_2)$  (shape-rate parameterization) and  $r_2 \sim \text{Gamma}(r_2; s_1, r_1)$  where  $(s_1, r_1, s_2)$  are parameters. The task we consider is to infer the posterior of the precision  $\tau$  of a normally distributed variable  $x \sim \mathcal{N}(x; 0, \tau)$  given realizations  $\{x_i\}_{i=1}^n$ . We consider the setting  $(s_1, r_1, s_2) = (1, 1, 1)$  which was used in Heess et al. (2013). Infer.NET’s implementation requires two gamma factors to specify the compound gamma. Here, we collapse them into one factor and let the operator learn to

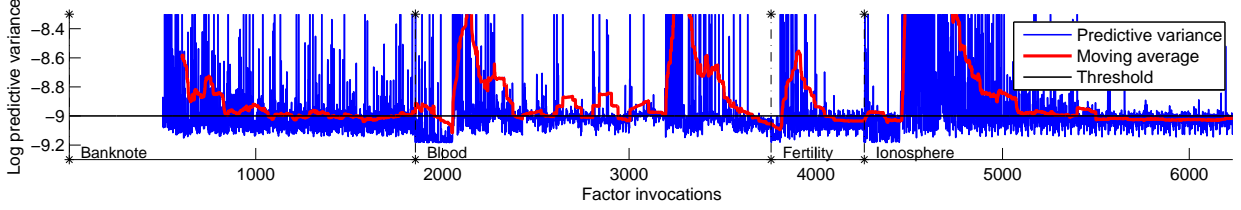


Figure 8: Uncertainty estimate of KJIT for outgoing messages on the four UCI datasets.

directly send an outgoing message  $m_{f \rightarrow \tau}$  given  $m_{\tau \rightarrow f}$ , using Infer.NET as the oracle. The default implementation of Infer.NET relies on a quadrature method. As in Eslami et al. (2014), we sequentially presented a number of problems to our algorithm, where at the beginning of each problem, a random number of observations  $n$  from 10 to 100, and the parameter  $\tau$ , were drawn from the model.

Fig. 7a and Fig. 7b summarize the inferred posterior parameters obtained from running only Infer.NET and Infer.NET + KJIT, i.e., KJIT with Infer.NET as the oracle. Fig. 7c shows the inference time of both methods. The plots collectively show that KJIT can deliver posteriors in good agreement with those obtained from Infer.NET, at a much lower cost. Note that in this task only one message is passed to the factor in each problem. Fig. 7c also indicates that KJIT requires fewer oracle consultations as more problems are seen.

**Experiment 5: Classification Benchmarks** In the final experiment, we demonstrate that our method for learning the message operator is able to detect changes in the distribution of incoming messages via its uncertainty estimate, and to subsequently update its prediction through additional oracle queries. The different distributions of incoming messages are achieved by presenting a sequence of different classification problems to our learner. We used four binary classification datasets from the UCI repository (Lichman, 2013): banknote authentication, blood transfusion, fertility and ionosphere, in the same binary logistic regression setting as before. The operator was required to learn just-in-time to send outgoing messages  $m_{f \rightarrow z_i}$  and  $m_{f \rightarrow p_i}$  on the four problems presented in sequence. The training observations consisted of 200 data points subsampled from each dataset by stratified sampling. For the fertility dataset, which contains only 100 data points, we subsampled half the points. The remaining data were used as test sets. The uncertainty threshold was set to -9, and the minibatch size was 500. All other parameters were the same as in the earlier JIT learning experiment.

Classification errors on the test sets and inference times are shown in Fig. 9a and Fig. 9b, respectively. The results demonstrate that KJIT improves the inference time on all the problems without sacrificing inference accuracy. The predictive variance of each outgoing message is shown in

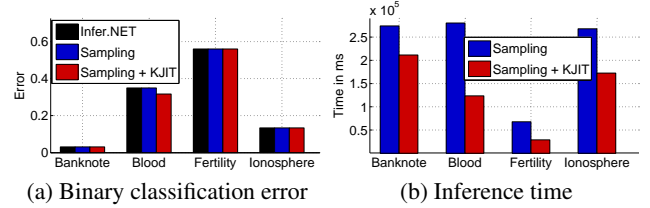


Figure 9: Classification performance and inference times on the four UCI datasets.

Fig. 8. An essential feature to notice is the rapid increase of the uncertainty after the first EP iteration of each problem. As shown in Fig. 1, the distributions of incoming messages of the four problems are diverse. The sharp rise followed by a steady decrease of the uncertainty is a good indicator that the operator is able to promptly detect a change in input message distribution, and robustly adapt to this new distribution by querying the oracle.

## 5 CONCLUSIONS AND FUTURE WORK

We have proposed a method for learning the mapping between incoming and outgoing messages to a factor in expectation propagation, which can be used in place of computationally demanding Monte Carlo estimates of these updates. Our operator has two main advantages: it can reliably evaluate the uncertainty of its prediction, so that it only consults a more expensive oracle when it is uncertain, and it can efficiently update its mapping online, so that it learns from these additional consultations. Once trained, the learned mapping performs as well as the oracle mapping, but at a far lower computational cost. This is in large part due to a novel two-stage random feature representation of the input messages. One topic of current research is hyperparameter selection: at present, these are learned on an initial mini-batch of data, however a better option would be to adapt them online as more data are seen.

## ACKNOWLEDGEMENT

We thank the anonymous reviewers for their constructive comments. WJ, AG, BL, and ZS thank the Gatsby Charitable Foundation for the financial support.

## References

- M. A. Alvarez, L. Rosasco, and N. D. Lawrence. Kernels for vector-valued functions: a review. 2011. URL <http://arxiv.org/abs/1106.6251>.
- S. Barthelmé and N. Chopin. ABC-EP: Expectation propagation for likelihood-free Bayesian computation. In *ICML*, pages 289–296, 2011.
- C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.
- A. Christmann and I. Steinwart. Universal kernels on non-standard input spaces. In *NIPS*, pages 406–414, 2010.
- A. Criminisi and J. Shotton. *Decision Forests for Computer Vision and Medical Image Analysis*. Springer Publishing Company, Incorporated, 2013.
- B. Dai, B. Xie, N. He, Y. Liang, A. Raj, M. Balcan, and L. Song. Scalable kernel methods via doubly stochastic gradients. In *NIPS*, pages 3041–3049, 2014.
- S. M. A. Eslami, D. Tarlow, P. Kohli, and J. Winn. Just-In-Time Learning for Fast and Flexible Inference. In *NIPS*, pages 154–162, 2014.
- A. Gelman. Prior distributions for variance parameters in hierarchical models. *Bayesian Analysis*, 1:1–19, 2006.
- N. Goodman, V. Mansinghka, D. Roy, K. Bonawitz, and J. Tenenbaum. Church: A language for generative models. In *UAI*, pages 220–229, 2008.
- N. Heess, D. Tarlow, and J. Winn. Learning to pass expectation propagation messages. In *NIPS*, pages 3219–3227, 2013.
- F. Hutter. *Automated Configuration of Algorithms for Solving Hard Computational Problems*. PhD thesis, University of British Columbia, Department of Computer Science, Vancouver, Canada, October 2009. <http://www.cs.ubc.ca/~hutter/papers/Hutter09PhD.pdf>.
- B. Lakshminarayanan, D. Roy, and Y.-W. Teh. Mondrian forests: Efficient online random forests. In *NIPS*, pages 3140–3148, 2014.
- Q. Le, T. Sarlós, and A. Smola. Fastfood - approximating kernel expansions in loglinear time. *ICML, JMLR W&CP*, 28:244–252, 2013.
- M. Lichman. UCI machine learning repository, 2013. URL <http://archive.ics.uci.edu/ml>.
- T. Minka, J. Winn, J. Guiver, S. Webster, Y. Zaykov, B. Yangel, A. Spengler, and J. Bronskill. Infer.NET 2.6, 2014. Microsoft Research Cambridge. <http://research.microsoft.com/infernet>.
- T. P. Minka. *A Family of Algorithms for Approximate Bayesian Inference*. PhD thesis, Massachusetts Institute of Technology, 2001. <http://research.microsoft.com/en-us/um/people/minka/papers/ep/minka-thesis.pdf>.
- A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *NIPS*, pages 1177–1184, 2007.
- C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006.
- W. Rudin. *Fourier Analysis on Groups: Interscience Tracts in Pure and Applied Mathematics, No. 12*. Literary Licensing, LLC, 2013.
- B. Schölkopf and A. J. Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. Adaptive computation and machine learning. MIT Press, 2002.
- A. Smola, A. Gretton, L. Song, and B. Schölkopf. A Hilbert space embedding for distributions. In *ALT*, pages 13–31, 2007.
- L. Song, A. Gretton, and C. Guestrin. Nonparametric tree graphical models. *AISTATS, JMLR W&CP*, 9:765–772, 2010.
- L. Song, A. Gretton, D. Bickson, Y. Low, and C. Guestrin. Kernel belief propagation. *AISTATS, JMLR W&CP*, 10:707–715, 2011.
- Stan Development Team. Stan: A C++ library for probability and sampling, version 2.4, 2014. URL <http://mc-stan.org/>.
- Z. Szabó, B. Sriperumbudur, B. Póczos, and A. Gretton. Learning theory for distribution regression. Technical report, Gatsby Unit, University College London, 2014. (<http://arxiv.org/abs/1411.2066>).
- D. Wingate, N. Goodman, A. Stuhlmüller, and J. Siskind. Nonstandard interpretations of probabilistic programs for efficient inference. In *NIPS*, pages 1152–1160, 2011.
- Z. Yang, A. J. Smola, L. Song, and A. G. Wilson. Á la carte - learning fast kernels. In *AISTATS*, 2015. <http://arxiv.org/abs/1412.6493>.



---

# Averaging of Decomposable Graphs by Dynamic Programming and Sampling

---

Kustaa Kangas      Teppo Niinimäki      Mikko Koivisto  
University of Helsinki, Department of Computer Science,  
Helsinki Institute for Information Technology HIIT, Finland  
{jwkangas,tzniinim,mkhkoivi}@helsinki.fi

## Abstract

We give algorithms for Bayesian learning of decomposable graphical models from complete data. We build on a recently proposed dynamic programming algorithm that finds optimal graphs of  $n$  nodes in  $O(4^n)$  time and  $O(3^n)$  space (Kangas et al., NIPS 2014), and show how it can be turned into accurate averaging algorithms. Specifically, we show that certain marginals of the posterior distribution, like the posterior probability of an edge, can be computed in  $O(n^3 3^n)$  time, provided that the prior over the graphs is of an appropriate form. To overcome some limitations of the exact approach, we also give sampling schemes that—using essentially no extra space—can draw up to  $3^n$  independent graphs from the posterior in  $O(n 4^n)$  time. Through importance sampling, this enables accurate Bayesian inference with a broader class of priors. Using benchmark datasets, we demonstrate the method’s performance and the advantage of averaging over optimization when learning from little data.

## 1 INTRODUCTION

A *decomposable graphical model* represents conditional independence relations between a set of variables by an undirected graph that is *decomposable*, or equivalently, *triangulated* or *chordal*. Due to their various nice properties—in particular, convenient parameterization, straightforward parameter learning from complete data, and computationally efficient inference—decomposable models have played a central role in both methodological and applied works (Lauritzen and Spiegelhalter 1988, Dawid and Lauritzen 1993, Abel and Thomas 2011).

It is common that the modeller hesitates to fix any specific graph, and would rather like to learn the graph from data. Learning the graph has, however, proved to be computationally very challenging (Srebro 2003, Corander et al. 2013).

The Bayesian approach, in particular, requires us to turn a prior into a posterior over all possible decomposable graphs. Or more practically, the interest is in drawing a representative (random) sample from the posterior or summarizing the posterior by some of its marginals. Solving these tasks by exhaustive enumeration of all graphs is feasible only up to about eight nodes. To manage with larger graphs, several Markov chain Monte Carlo (MCMC) methods have been proposed (Madigan and York 1995, Giudici and Green 1999, Tarantola 2004, Corander et al. 2006, Green and Thomas 2013). While these methods may work well in many cases, they offer essentially no guarantees concerning the accuracy of the produced estimates.

In this paper, we present exact averaging and sampling algorithms that admit Bayesian learning of decomposable graphs up to about 20 nodes. We build on the recent dynamic programming (DP) algorithm of Kangas et al. (2014). Their algorithm finds a decomposable graph that *maximizes* a given decomposable scoring function, for example, the posterior probability. For graphs with  $n$  nodes the algorithm takes  $O(4^n)$  time and  $O(3^n)$  space. Generally, it is straightforward to turn a DP algorithm, designed for an optimization problem, into a “corresponding” averaging or sampling algorithm (and vice versa). At first glance, our result may thus look nothing but a rephrasing of the result of Kangas et al.

However, this view turns out to be only partial. Indeed, replacing maximization by averaging opens new algorithmic opportunities, which allow us to give an asymptotically faster,  $O(n^3 3^n)$ -time algorithm (Sect. 3). On the other hand, the averaging viewpoint also brings new difficulties: First, it turns out that the particular DP treatment applies to Bayesian model averaging only when the prior over the graphs is of a specific form; for example, the form cannot express the uniform prior over all decomposable graphs, whereas it can express the prior that is proportional to the number of so-called rooted junction trees of the graph. Second, it appears that exact computations are feasible only for marginal posterior probabilities of small subgraphs, like of an individual edge, but not of large subgraphs, not to mention more

complicated graph features. We overcome these difficulties by designing schemes that, using the computed DP tables, can efficiently generate large numbers of random samples from the posterior or its computationally convenient proxy (Sect. 4). We then feed the samples to usual Monte Carlo estimators of the quantities of interest (Sect. 5).

We demonstrate the performance of the methods using benchmark datasets (Sect. 6). First, we investigate whether averaging over graphs yields significantly better prediction results as compared to using a single maximum-a-posteriori graph. Second, we study the accuracy of the sampling based estimators for edge posterior probabilities.

Finally, we discuss straightforward ways to further enhance the presented methods, and also mention some major open questions (Sect. 7).

## 2 PRELIMINARIES

We review some fundamentals of decomposable models. For a more thorough treatment, see Lauritzen (1996).

### 2.1 DECOMPOSABILITY AND LEARNING

Consider an undirected graph  $G$  on a set of nodes  $V = \{1, \dots, n\}$ . We call a subset of nodes *complete* if it induces a complete subgraph. Further, we call a complete set a *clique* if it is *maximal*, that is, not a subset of any other complete set. We denote the set of cliques of  $G$  simply by  $\mathcal{C}$ , assuming there is no ambiguity about the referred graph.

We call  $G$  *decomposable* if it has the *running intersection property*, that is, there is an ordering of its cliques,  $C_1, \dots, C_k$ , such that for each  $i = 2, \dots, k$  we have that  $S_i = (C_1 \cup \dots \cup C_{i-1}) \cap C_i \subset C_j$  for some  $j < i$ . The sets  $S_i$  are called the *separators* and they form a multiset, which we denote by  $\mathcal{S}$ . We note that  $\mathcal{S}$  is uniquely specified by  $G$  and does not depend on the ordering of the cliques.

Suppose  $G$  is decomposable and with each node  $v \in V$  associate a random variable  $x_v$ . The graph  $G$  together with a joint distribution  $p$  over the variables form a *decomposable graphical model* if  $p$  factorizes as

$$p(x_V) = \frac{\prod_{C \in \mathcal{C}} p(x_C)}{\prod_{S \in \mathcal{S}} p(x_S)},$$

where we write  $x_S$  for the tuple  $(x_v : v \in S)$ . The factorization plays a central role in probabilistic inference in a given decomposable model, and in learning a model for a fixed graph from *data*, that is, multiple records over the variables.

The factorization is central also when learning the graph. In the Bayesian approach to learning we turn a *prior*  $\rho$  over the graphs into a *posterior*  $\pi$  by multiplying the prior by the (marginal) *likelihood*  $\ell$ , and dividing by the *normalizing constant* of the function  $\rho\ell$ , given as  $Z_{\rho\ell} = \sum_G \rho(G)\ell(G)$ .

The likelihood  $\ell(G)$  is the probability (density) of the data, given  $G$ . It is obtained by integrating out the parameters that specify the distribution  $p$  above. Under commonly adopted parameter priors (Dawid and Lauritzen 1993), the likelihood function factorizes into a product of local marginal likelihoods, one term per clique and separator. Thus the likelihood function is an example of a decomposable function:

**Definition 1** (decomposable function). Let  $V$  be a finite set and let  $\varphi$  be a function from the decomposable graphs on  $V$  to real numbers. We say that  $\varphi$  is *decomposable* over  $V$  if

$$\varphi(G) = \frac{\prod_{C \in \mathcal{C}} \varphi_c(C)}{\prod_{S \in \mathcal{S}} \varphi_s(S)}$$

for some functions  $\varphi_c$  and  $\varphi_s$ , we call the *local components*.

Clearly the product of two decomposable functions is also decomposable. It can also be shown that any constant function is decomposable. In particular, if the prior is a decomposable function, so is the posterior.

We illustrate the notion of decomposable functions by two further examples. Here and henceforth we use the Iverson bracket  $[Q]$  to denote 1 when  $Q$  is true, and 0 otherwise.

**Example 1** (uniform prior). Let  $w$  be a number. Let  $\rho$  be the decomposable function over  $V$  defined by

$$\rho_c(X) = [ |X| \leq w ] \quad \text{and} \quad \rho_s(X) = 1 \quad \text{for } X \subseteq V.$$

We observe that  $\rho(G) = 1$  if  $G$  contains only cliques of size at most  $w$ , and 0 otherwise. Thus the normalized function  $\rho/Z_\rho$  is the uniform distribution over the decomposable graphs on  $V$  whose cliques are of size at most  $w$ .

**Example 2** (absence of an edge). Let  $e \subseteq V$ ,  $|e| = 2$ . Let  $\varphi^e$  be the decomposable function over  $V$  defined by

$$\varphi_c^e(X) = [e \not\subseteq X] \quad \text{and} \quad \varphi_s^e(X) = 1 \quad \text{for } X \subseteq V.$$

We observe that  $\varphi^e(G) = 1$  if the edge  $e$  is absent in  $G$ , and 0 otherwise. Thus  $\varphi^e$  is the indicator function of the set of decomposable graphs that do not contain the edge  $e$ .

Note also that the indicator function for the *presence* of an edge is not decomposable.

### 2.2 COMPUTATIONAL TASKS

We will consider two classes of computational problems: (1) computing the *marginal* of a given function  $\varphi$  of decomposable graphs, defined as  $Z_\varphi = \sum_G \varphi(G)$ ; and (2) generating random samples of decomposable graphs from a distribution that is proportional to a given function. The first class includes the important task of computing posterior expectations of graph features, in particular, marginal posterior probabilities of edges (cf. Corollary 2 in Sect. 3). In the second class the distribution of interest is usually the posterior distribution. The input in these problems is always specified by decomposable functions, like a prior and a likelihood function. Thus we may assume an efficient access to these functions through their local components.

### 2.3 ROOTED JUNCTION TREES

It is convenient to represent a decomposable graph as a *junction tree*. Consider an undirected graph  $G$  and a tree  $\mathcal{J}$  that has the cliques of  $G$  as its vertices. We call  $\mathcal{J}$  a junction tree of  $G$  if it satisfies the *junction property*, that is, for any cliques  $C, C'$  the intersection  $C \cap C'$  is contained within every clique on the unique path between  $C$  and  $C'$  in  $\mathcal{J}$ . A graph has a junction tree if and only if it is decomposable. Importantly, the representation is in general not unique, as a graph may have multiple junction trees; we denote by  $\tau(G)$  the number of junction trees of  $G$ . In contrast, for each junction tree  $\mathcal{J}$  the represented graph,  $G(\mathcal{J})$ , is unique.

For our purposes it will be convenient to work with rooted junction trees. Specifically, we make use of the following recursive characterization of junction trees, adopted from the work of Kangas et al. (2014):

**Definition 2** (recursive partition tree, RPT). A *recursive partition tree* over a finite ground set  $V$  is a triplet  $(C, \{R_1, \dots, R_k\}, \{\mathcal{T}_1, \dots, \mathcal{T}_k\})$  such that

1.  $C$  is a non-empty subset of  $V$ , called the *root*;
2.  $\{R_1, \dots, R_k\}$  is a partition of  $V \setminus C$ ;
3. each  $\mathcal{T}_i$  is an RPT over  $C \cup R_i$  rooted at  $C_i$  such that  $C \cap C_i$  is a proper subset of both  $C$  and  $C_i$ .

Note that the implicit base case of the definition is when  $C = V$  and there are thus no subtrees.

An RPT  $\mathcal{T}$  rooted at  $C$  can be viewed as a directed tree, where  $C_1, \dots, C_k$  are the children and  $\mathcal{T}_1, \dots, \mathcal{T}_k$  the subtrees of  $C$ . Such a tree is a junction tree and, conversely, any junction tree can be represented as an RPT, which is unique up to the choice of the root (Kangas et al. 2014). Denoting by  $\kappa(G)$  the number of cliques of a decomposable graph  $G$ , we have that  $G$  has exactly  $\tau(G)\kappa(G)$  distinct RPTs.

### 2.4 DYNAMIC PROGRAMMING

Kangas et al. considered the problem of maximizing a given decomposable function, and gave a DP algorithm that stems from the recursive definition of RPTs. We next adapt the DP algorithm to the problem of summing up the values of a given function, that is, to compute the marginal. Due to the many-to-one relationship of RPTs and decomposable graphs, each value gets multiplied by the corresponding number of RPTs. Put otherwise, for any function  $\varphi$  we have

$$\sum_{\mathcal{T}} \varphi(G(\mathcal{T})) = \sum_G \varphi(G)\tau(G)\kappa(G), \quad (1)$$

where  $\mathcal{T}$  runs through all RPTs over  $V$  and  $G$  runs through all decomposable graphs on  $V$ . Since our DP treatment relies on the decomposability of the function  $\varphi$ , the computed sum will not be the marginal of the decomposable  $\varphi$ , but of the function  $\varphi\tau\kappa$ , that we shall call RPT-decomposable:

**Definition 3** (RPT-decomposable function). Let  $V$  be a finite set and let  $\varphi'$  be a function from the decomposable graphs on  $V$  to real numbers. We say that  $\varphi'$  is *RPT-decomposable* over  $V$  if  $\varphi'(G) = \varphi(G)\tau(G)\kappa(G)$  for some decomposable function  $\varphi$  over  $V$ .

**Example 3** (RPT-uniform prior). Let  $\rho$  be as defined in Example 1. Then  $\rho\tau\kappa$  is a RPT-decomposable function that is proportional to the uniform distribution on all RPTs over  $V$  whose cliques are of size at most  $w$ .

We are ready to present the DP algorithm for computing the sum (1) for a given a decomposable function  $\varphi$ . To this end, we denote by  $\text{RPT}(S, R)$  the set of all RPTs over  $S \cup R$  rooted at a proper superset of  $S$ , and let

$$f(S, R) = \sum_{\mathcal{T} \in \text{RPT}(S, R)} \varphi(G(\mathcal{T})).$$

In particular,  $f(\emptyset, V)$  equals the desired sum (1). Following Kangas et al. we obtain the following recurrence system:

$$f(S, R) = \sum_{S \subset C \subseteq S \cup R} \varphi_c(C) g(C, R \setminus C), \quad (2)$$

$$g(C, U) = \sum_{\min U \in R \subseteq U} h(C, R) g(C, U \setminus R), \quad (3)$$

$$h(C, R) = \sum_{S \subset C} f(S, R) / \varphi_s(S), \quad (4)$$

with the base case  $g(C, \emptyset) = 1$ . Each recurrence is defined for all disjoint pairs of subsets of  $V$  such that  $C$  and  $R$  are non-empty. A straightforward evaluation of  $f$ ,  $g$ , and  $h$  by using these recurrences takes  $O(4^n)$  time and  $O(3^n)$  space. These bounds and the correctness of the recurrences can be verified essentially by taking the proof of Kangas et al. and replacing maximization with summation.

The intuition here is that  $f$  considers all possible choices for the root clique  $C$  and factors in the  $\varphi_c(C)$ . For each  $C$  it then invokes  $g$ , which considers partitions  $\{R_1, \dots, R_k\}$  of the remaining nodes  $U$  by calling itself recursively. For each part  $R$  it also calls  $h$ , which considers possible separators  $S$  between  $C$  and the subtree in  $R$  and factors in the reciprocal of the marginals  $\varphi_s(S)$ . Finally,  $h$  calls  $f$  again to consider possible root cliques of the subtree. In (3), the least element of  $U$ , denoted  $\min U$ , is always placed in the next part  $R$  so as not to consider different permutations of the same partition. The base case corresponds to the case where all nodes have been assigned to some  $R_i$  and there are none left to partition.

### 3 EXACT AVERAGING

In this section we show:

**Theorem 1.** *The marginal of a given RPT-decomposable function over a set of  $n$  nodes can be computed in  $O(n^3 3^n)$  time and  $O(n 3^n)$  space.*

In effect, we show that the recurrence system (2–4) can be solved in the claimed time. Asymptotically, this is significantly faster than the  $O(4^n)$  time obtained by a straightforward evaluation. We achieve the improvement by computing the exponential-size summations for each of the functions  $f$ ,  $g$ , and  $h$  simultaneously for several pairs of arguments. Our algorithms for  $f$  and  $h$  apply as well to the maximization variant of the recurrence. In contrast, our algorithm for  $g$  relies crucially on subtraction (i.e., the existence of additive inverses), and thus does not apply to maximization.

Before we proceed to the proof (in Sections 3.1–3.5), let us note the following implication:

**Corollary 2.** *Suppose the probability distribution over decomposable graphs on a set of  $n$  nodes is proportional to a given RPT-decomposable function. Then the probability that the graph contains  $k$  specified edges can be computed in  $O(2^k n^3 3^n)$  time and  $O(n 3^n)$  space.*

*Proof.* Let  $e_1, \dots, e_k$  be distinct edges on the node set  $V = \{1, \dots, n\}$ . Let  $A_j$  denote the event that the graph does *not* contain the edge  $e_j$ . By the inclusion–exclusion principle, the probability that the graph contains the  $k$  edges is

$$\Pr[\bar{A}_1 \cap \dots \cap \bar{A}_k] = \sum_{J \subseteq \{1, \dots, k\}} (-1)^{|J|} \Pr\left[\bigcap_{j \in J} A_j\right].$$

Now, let  $\varphi'$  be the given RPT-decomposable function. It remains to observe that

$$\Pr\left[\bigcap_{j \in J} A_j\right] = \frac{\sum_G \varphi'(G) \prod_{j \in J} \varphi^{e_j}(G)}{\sum_G \varphi'(G)},$$

where each  $\varphi^{e_j}$  is the decomposable function that indicates whether  $e_j$  is absent in the graph, as defined in Example 2. Thus each of the  $2^k$  probabilities is obtained as a ratio of two marginals of RPT-decomposable functions.  $\square$

### 3.1 ZETA TRANSFORM AND SUBSET CONVOLUTION

Our algorithms employ the so-called *fast zeta transform*, FZT (Yates 1937, Kennes and Smets 1990, Björklund et al. 2012). The *zeta transform* of a function  $\alpha$  from the subsets of a ground set  $\{1, \dots, n\}$  to real numbers is the set function defined by  $\hat{\alpha}(Y) = \sum_{X \subseteq Y} \alpha(X)$  for each subset  $Y$  of the ground set. FZT computes the zeta transform of a given function in  $O(n 2^n)$  time, as follows: Let  $\alpha_0 = \alpha$  and for  $i = 1, \dots, n$  let

$$\alpha_i(Y) = \alpha_{i-1}(Y) + [i \in Y] \cdot \alpha_{i-1}(Y \setminus \{i\}).$$

It follows that  $\alpha_n = \hat{\alpha}$ . Note that there are two different ways to organize the computations: either compute the  $n$  steps one after another, which requires only  $O(2^n)$  space; or

compute all the  $n$  functions for each set  $Y$  one after another in increasing order by the size  $|Y|$ , which requires  $O(n 2^n)$  space. We will need the latter “level-wise” implementation.

Another tool we use is known as the *fast subset convolution*, FSC (Björklund et al. 2007). The *subset convolution* of two functions  $\alpha$  and  $\beta$  from the subsets of a ground set  $\{1, \dots, n\}$  to real numbers is the set function defined by  $(\alpha * \beta)(Y) = \sum_{X \subseteq Y} \alpha(X) \beta(Y \setminus X)$  for each subset  $Y$  of the ground set. FSC computes the subset convolution in  $O(n^2 2^n)$  time and  $O(n 2^n)$  space. We refer to Björklund et al. (2007) for details.

### 3.2 COMPUTING $h$

Consider first the recurrence for  $h$ . For a moment, fix a set  $R \subseteq V$  and define the functions  $h_R$  and  $f_R$  by

$$h_R(C) = h(C, R) \quad \text{and} \quad f_R(S) = f(S, R) / \varphi_s(S),$$

where  $C$  and  $S$  are subsets of  $V \setminus R$ . By the recurrence (4) we have that  $h_R(C) = \sum_{S \subseteq C} f_R(S) = \hat{f}_R(C) - f_R(C)$ . Using FZT we can compute  $h_R$  in  $O(k 2^k)$  time for each  $R$ , where  $k = n - |R|$ . Note that we use the level-wise implementation and evaluate the  $k$  steps of the transform for one  $C$  in turn. Thus computing  $h$  takes  $O(n 3^n)$  time in total. (While we here used subtraction, it is not difficult to see how FZT can be modified to avoid that, and that the result thus applies to the maximization variant as well.)

### 3.3 COMPUTING $f$

Consider then the recurrence for  $f$ . It might be tempting to try the above trick also in this case, that is, to fix either  $R$  or  $S$ , and then employ a suitable fast zeta transform variant. However, that approach fails because now the involved three sets  $S$ ,  $R$ , and  $C$  have more intricate dependencies. Instead, we define the function  $g'$  by

$$g'(C, R \setminus C) = \varphi_c(C) g(C, R \setminus C),$$

for  $C \subseteq R \subseteq V$ . The idea is to extend FZT to pairs of disjoint sets and transform  $g'$  into  $f$  by computing the sum (2) in  $n$  steps: Let  $g'_0 = g'$  and for  $i = 1, \dots, n$  let

$$g'_i(S, R) = g'_{i-1}(S, R) + [i \in R] \cdot g'_{i-1}(S \cup \{i\}, R \setminus \{i\}).$$

It can be shown by simple induction that  $g'_n(S, R) = f(S, R) + g'(S, R \setminus S)$  (see the supplement). Thus, given  $g'$ , we can compute  $f$  in  $O(n 3^n)$  time. Note that we use the level-wise approach to compute the values  $f(S, R)$  in *decreasing* order of  $|S|$  and in *increasing* order of  $|R|$ .

### 3.4 COMPUTING $g$

Finally consider the recurrence for  $g$ . Now we fix a  $C \subseteq V$  for a moment and, for convenience, write  $g_C(U)$  for  $g(C, U)$

and  $h_C(R)$  for  $h(C, R)$ . We will show that the values

$$g_C(U) = \sum_{\min U \in R \subseteq U} h_C(R) g_C(U \setminus R)$$

can be computed for all  $U \subseteq V$  of size  $|U| = u$  in  $O(n^2 2^n)$  time, assuming the values  $h_C(R)$  and  $g_C(U')$  are available for all  $R, U' \subseteq V$  of sizes  $|R| \leq u$  and  $|U'| < u$ . This then implies that computing  $g$  takes  $O(n^3 3^n)$  time in total (by summing over  $u$  and  $C$ ).

To compute the values  $g_C(U)$ , we break the computations further into  $n$  separate subtasks. For each  $s \in V$ , define the function  $g^s$  by

$$g^s(U) = \sum_{s \in R \subseteq U} h(R) g(U \setminus R),$$

where  $U \subseteq V$  such that  $|U| = u$  and  $\min U = s$ . Note that each such  $U$  is of the form  $\{s\} \cup X$  with  $X \subseteq V \setminus \{1, \dots, s\}$ . We observe that for each  $s$  the task can be solved using FSC in  $O(n^2 2^{n-s})$  time, implying a time requirement of  $O(n^2 2^n)$  in total. It remains to observe that  $g(U)$  is obtained as  $g^{\min U}(U)$ .

### 3.5 SPACE REQUIREMENT

The level-wise computations of  $h$  and  $f$  assume that the intermediate values of the transforms are stored in memory. This incurs a space requirement of  $(n 3^n)$  in total.

The computation of  $g$ , instead, uses FSC in a black-box fashion and computes a bunch of values  $g(C, U)$  for a fixed  $C$  and several sets  $U$  of a fixed size  $u$ , assuming only that the values  $h(C, R)$  and  $g(C, U')$  are available for all  $R$  and  $U'$  such that  $|R| \leq u$  and  $|U'| < u$ . Because the same space can be reused for different  $C$ , the “extra” space requirement is only that of FSC,  $O(n 2^n)$ .

## 4 SAMPLING

We now turn to the task of sampling decomposable graphs from the posterior. Specifically, we extend the DP algorithm of Sect. 2 with a natural backtracking procedure that draws an RPT by choosing its cliques, partitions, and separators according to their conditional marginal probabilities, and returns the corresponding graph. Since each graph  $G$  has  $\tau(G)\kappa(G)$  distinct RPTs, the resulting graph sample follows the RPT-decomposable distribution  $\pi \propto \varphi \tau \kappa$ , where  $\varphi$  is the decomposable function that the DP algorithm was applied to. By employing importance sampling (Sect. 5) that weights each graph sample  $G$  by  $(\tau(G)\kappa(G))^{-1}$ , we are able to target the distribution proportional to  $\varphi$  instead.

For the remainder of this section we focus on sampling from  $\pi \propto \varphi \tau \kappa$ . Given the DP tables for  $f$ ,  $g$ , and  $h$  (for  $\varphi$ ), the backtracking procedure to sample from  $\pi$  works as follows.

### Algorithm: Sampling a decomposable graph

Begin by visiting  $f(\emptyset, V)$  and recursively visit  $f$ ,  $g$  and  $h$  as follows: In  $f(S, R)$ , choose the root clique  $C$  of the subtree over  $S \cup R$  randomly according to its marginal distribution  $\Pr(C) \propto \varphi_c(C) g(C, R \setminus C)$  and proceed to corresponding  $g(C, R \setminus C)$ . In  $g(C, U)$ , choose the first part  $R$  of the partition according to its marginal distribution  $\Pr(R) \propto h(C, R) g(C, U \setminus R)$  and recursively proceed to  $h(C, R)$  and  $g(C, U \setminus R)$ . In  $h(C, R)$ , choose a separator  $S$  according to its marginal probability  $\Pr(S) \propto f(S, R) / \varphi_s(S)$  and proceed to  $f(S, R)$ . Continue the recursion until all branches terminate at a visit to  $g(C, \emptyset)$ . Then, from the resulting recursion tree, obtain the cliques  $C$  and return the corresponding graph.

The efficiency of this backtracking procedure depends on how much time is spent on choosing random sets  $C$ ,  $R$ , and  $S$  during the visits. Consider the general problem of drawing a sample from a discrete distribution over  $s$  elements. A naive method is to pick a number  $r$  from the continuous uniform distribution on  $[0, 1)$ , then iterate the elements in a predefined order and select the first item for which the cumulative probability exceeds  $r$ . The method uses  $O(s)$  time and  $O(1)$  space. It can be shown that, if the sets  $C$ ,  $R$ , and  $S$  are sampled using this naive method, sampling a single graph requires  $O(2^n)$  time. (In fact, this result is proved as a special case in the next subsection.) Next we introduce a more general sampling scheme that allows us to spend less time in sampling by using more space.

### 4.1 TRADING TIME FOR SPACE

We can avoid spending exponential time per sample by first preprocessing the probabilities so that the sets can thereafter be chosen much faster. First, it should be noted that it is not possible to just precompute the cumulative probabilities and use, for example, binary search to find the item that corresponds to  $r$ , without increasing the asymptotic space requirement. On the other hand, if we allow additional space usage, then it is better to use the *alias method* (Vose 1991), which requires  $O(s)$  additional space and  $O(s)$  preprocessing time but allows us to draw samples in  $O(1)$  time per sample. Moreover, it turns out we obtain a tunable tradeoff between sampling time and extra space. Specifically, we propose the following algorithm:

#### Subroutine: Parameterized sampling

Let  $b \in \{0, \dots, n\}$  be a tradeoff parameter that is chosen beforehand. As preprocessing, divide the  $s$  elements into bins of size  $2^b$  (at most, the last bin can be smaller), and for each bin, compute and store the sum of the probabilities of its elements. This requires space that is linear in the number of bins.

Now a new sample can be drawn in two phases: first a bin is selected according to its precomputed total proba-

bility and then a term is selected from the bin according to its relative probability in the bin. We apply the alias method to the first phase, and the naive sampling to the second phase. Thus, the first phase requires  $O(1)$  time per sample. The additional space requirement is linear in the number of bins. The second phase uses  $O(2^b)$  time and no additional space. The total space requirement is thus  $O(s/2^b)$  and time requirement is  $O(s)$  for preprocessing and  $O(2^b)$  per sample.

By selecting a fixed  $b$  and applying the above algorithm to each (nonterminating)  $f(S, R)$ ,  $g(C, U)$ , and  $h(C, R)$  we get the following theorem:

**Theorem 3.** *For any  $b \in \{0, \dots, n\}$  we can draw  $T$  independent graphs from  $\pi$  in  $O(4^n + T \cdot 2^b(1 + n - b))$  time and  $O(4^n/2^b + 3^n)$  space.*

In order to prove the theorem, we first bound the number of visits to  $f$ ,  $g$ , and  $h$  by the following lemma:

**Lemma 4.** *The sampling procedure makes at most  $n$  nonterminating visits to functions  $f$ ,  $g$ , and  $h$  each.*

The proof of Lemma 4 is given in the supplement.

Now, by using Lemma 4 and the fact that on each visit to  $f$ ,  $g$  or  $h$ , the time required to choose the set  $C$ ,  $R$  or  $S$  correspondingly is at most  $O(2^b)$ , we get a bound  $O(2^b n)$  for the time consumption per sample. However, in order to get the bound down to  $O(2^b(1+n-b))$  as in Theorem 3, we need to bound the amount of work done on each visit more carefully. To this end, observe that each nonterminating visit to  $f(S, R)$ ,  $g(C, U)$ , and  $h(C, R)$  involves drawing a random set from a discrete distribution over  $2^{|R|} - 1 < 2^{|S|+|R|}$ ,  $2^{|U|}/2 < 2^{|C|+|U|}$ , and  $2^{|C|} - 1 < 2^{|C|+|R|}$  sets respectively. The following lemma bounds  $|S| + |R|$ ,  $|C| + |U|$ , and  $|C| + |R|$  on different steps of the backtracking.

**Lemma 5.** *For  $f$ ,  $g$ , and  $h$  let  $(S_1, R_1), \dots, (S_{d_f}, R_{d_f})$ ,  $(C_1, U_1), \dots, (C_{d_g}, U_{d_g})$ , and  $(C_1, R_1), \dots, (C_{d_h}, R_{d_h})$  be all the set pairs visited during backtracking. Then there exist orderings of those set pairs such that,*

$$|S_i| + |R_i| \leq n - i + 1 \quad \text{for all } i = 1, \dots, d_f,$$

$$|C_i| + |U_i| \leq n - i + 1 \quad \text{for all } i = 1, \dots, d_g,$$

$$|C_i| + |R_i| \leq n - i + 1 \quad \text{for all } i = 1, \dots, d_h.$$

The proof of Lemma 5 is given in the supplement.

Now we are ready to prove the theorem presented above.

*Proof of Theorem 3.* Consider the space and time needed for  $f$  (respectively:  $g, h$ ).

**Space:** For each  $R$  (respectively:  $U, C$ ) of size  $k$  there are  $2^{n-k}$  ways to choose  $S$  (respectively:  $C, R$ ). The space needed by the alias method for each such set pair is less

than  $\max\{2^k/2^b, 1\}$ . Thus the total space requirement for all set pairs of all sizes is

$$\sum_k \binom{n}{k} 2^{n-k} \max\{2^{k-b}, 1\} \leq \sum_k \binom{n}{k} (2^{n-b} + 2^{n-k}) = \frac{4^n}{2^b} + 3^n.$$

**Time:** The preprocessing requires enumeration over all terms of the sums in the recurrences for all entries of  $f$ ,  $g$ , and  $h$ . Like computing  $f$ ,  $g$ , and  $h$ , this consumes  $O(4^n)$  time. It remains to analyze the time needed to draw a single graph sample. Let  $(S_1, R_1), \dots, (S_d, R_d)$  (respectively:  $(R_1, U_1), \dots, (R_d, U_d)$ ,  $(C_1, R_1), \dots, (C_d, R_d)$ ) be the  $d$  visited set pairs for  $f$  (respectively:  $g, h$ ). In each visit the algorithm first chooses a bin in constant time and then chooses one of its elements in time linear in its size but at most  $2^b$ . Therefore, the backtracking requires at most time

$$\sum_{i=1}^d \min\{2^b, 2^{k_i}\},$$

where  $k_i = |R_i|$  (respectively:  $k_i = |U_i|$ ,  $k_i = |C_i|$ ). By Lemma 5, there exists an ordering of the set pairs such that  $k_i \leq n - i + 1$ . By Lemma 4,  $d \leq n$ . We get

$$\begin{aligned} \sum_{j=1}^n \min\{2^b, 2^j\} &\leq \sum_{j=b+1}^n 2^b + \sum_{j=1}^b 2^j \\ &\leq (n-b)2^b + 2^{b+1} \\ &= (2+n-b)2^b. \end{aligned}$$

The claim thus follows.  $\square$

Theorem 3 has, for example, the following corollaries. Setting  $b = 0$  allows us to draw each sample in linear time but using  $O(4^n)$  extra space, while setting  $b = n$  effectively yields the naive method. From an asymptotical viewpoint, it makes sense to set  $b \leq n \log_2(4/3) \simeq 0.42n$ , since for larger  $b$  the  $3^n$  term dominates the space requirement. Since the DP phase requires  $O(4^n)$  time, we can in a sense draw  $O(4^n/2^{n \log_2(4/3)}) = O(2^n/n)$  samples for “free.”

## 4.2 ADAPTIVE SAMPLING

Usually most of the probability mass is concentrated on a small set of graphs. Thus, when sampling graphs, some of the indexing set pairs for  $f$ ,  $g$ , and  $h$  are visited rarely if at all. The additional space that is used by the alias method for such set pairs may outweigh the gain in speed. As an alternative, we propose the following approach that periodically draws and caches multiple samples at once on those indices that are visited more often.

### Subroutine: Adaptive sampling

On the first visit to any  $f(S, R)$ ,  $g(C, U)$ , or  $h(C, R)$ , draw and consume one set from the corresponding discrete distribution using the naive method. On any subsequent visit: If there are no cached samples left from

the previous visits, then use the alias method to draw twice as many sets as the last time on the same index, consume one and cache the rest. Otherwise, consume one set from the cache.

The following lemma characterizes the space consumption of the graph sampler that uses the above adaptive sampling subroutine.

**Lemma 6.** *After  $T$  sampled graphs, adaptive sampling consumes at most  $O(nT)$  extra space.*

*Proof.* A cache of size  $s$  is constructed when the corresponding set pair is visited for the  $(s + 1)$ th time. By Lemma 4 the total number of visits to all set pairs is at most  $nT$ . The claim then trivially follows.  $\square$

Furthermore, we get the following special case bounds for space and time consumption:

**Theorem 7.** *We can draw  $3^n$  independent samples from  $\pi$  in  $O(n4^n)$  time and  $O(n3^n)$  space.*

*Proof.* The space complexity follows from Lemma 6.

Consider then the time used on visits to  $f$ .

Without rebuilding the caches, the time per sample is  $O(n)$ , or  $O(n3^n)$  in total. In order to analyze the time needed for cache rebuilds, consider an arbitrary disjoint set pair  $(S, R)$  and let  $k = |R|$ .

Let  $t$  be the number of visits to the set pair in question. Then its cache is rebuilt  $\lfloor \log_2(t) + 1 \rfloor$  times. As rebuilding the cache the  $i$ th time requires  $O(2^k + 2^i)$  time (the first term comes from constructing the distribution and initializing the alias method, the second term comes from drawing the sets), in total this requires time

$$\sum_{i=1}^{\lfloor \log_2(t)+1 \rfloor} O(2^k + 2^i) = O((\log(t) + 1)2^k) + O(t).$$

To get the total time used to rebuild caches, we must sum these for all set pairs  $(S, R)$ . Since each backtracking visits at most  $n$  set pairs (Lemma 4), the sum over the last term  $O(t)$  results in  $O(n3^n)$ . It remains to analyze the first term.

For  $k = 1, \dots, n$ , let  $L_k$  consist of all the set pairs  $(S, R)$  such that  $|R| = k$ . For a fixed  $k$ , there are  $|L_k| = \binom{n}{k} 2^{n-k}$  such pairs, and each may be visited at most once per sample, that is, at most  $3^n$  times in total. Thus, the sum of the first term over all pairs in  $L_k$  amounts to  $O(|L_k|(\log(3^n) + 1)2^k) = O\left(n \binom{n}{k} 2^n\right)$ . Summing over  $k$  yields the claimed running time bound.

The time used on visits to  $g$  and  $h$  can be bounded analogously (selecting  $k = |U|$  and  $k = |C|$ , respectively).  $\square$

## 5 MONTE CARLO ESTIMATION

We have observed in Sections 2 and 3 that the posterior probabilities of some graph properties can be computed exactly in  $O(4^n)$  time, and asymptotically even faster in  $O(n^3 3^n)$  time. However, we had to assume (i) that the properties can be expressed by decomposable functions (e.g., in terms of forbidden cliques), and (ii) that the prior, and hence the posterior, is RPT-decomposable, ruling out the natural uniform prior.

In this section, we employ Monte Carlo methods to relax these restrictive assumptions and still obtain good approximations. Throughout the section, we consider an arbitrary function  $\psi$  from decomposable graphs to  $\{0, 1\}$ . Our interest is in estimating the quantity

$$Z_{\pi\psi} = \sum_G \pi(G)\psi(G),$$

where  $\pi$  is the posterior distribution.

### 5.1 RPT-DECOMPOSABLE PRIOR

Consider first relaxing only the first assumption (i). Since we keep the assumption that the prior is RPT-decomposable, we can draw  $T$  independent graphs  $G_1, \dots, G_T$  from the posterior  $\pi$ , using the methods described in Sect. 4. The estimate

$$\hat{Z}_{\pi\psi} = \frac{1}{T} \sum_{i=1}^T \psi(G_i)$$

is unbiased and, by the law of large numbers, it concentrates around  $Z_{\pi\psi}$  as  $T$  grows.

### 5.2 DECOMPOSABLE PRIOR

Consider then relaxing also the second assumption (ii). For convenience, assume however that the prior is decomposable, for example, the uniform distribution over all decomposable graphs on the node set  $V$ . Now, we can draw  $T$  independent graphs  $G_1, \dots, G_T$  from a distribution proportional to  $\pi\tau\kappa$ , using the methods described in Sect. 4. In this case, we need to correct the deviance of the sampling distribution from the posterior. We do this by using the self-normalized importance sampling estimate

$$\tilde{Z}_{\pi\psi} = \frac{\sum_{i=1}^T w_i \psi(G_i)}{\sum_{i=1}^T w_i}, \quad w_i = \frac{1}{\tau(G_i)\kappa(G_i)}.$$

This estimate concentrates around  $Z_{\pi\psi}$  as  $T$  grows.

To make this method practical, we need an efficient way to count the number of junction trees  $\tau(G_i)$  and the number of cliques  $\kappa(G_i)$  of a given decomposable graph  $G_i$ . The latter problem is easy, as the cliques are readily available in the RPT (or junction tree) representation. For counting junction

Table 1: Benchmark datasets on  $n$  variables and  $m$  records.

| Dataset     | $n$ | $m$   |
|-------------|-----|-------|
| Asia        | 8   | 10000 |
| Bridges     | 12  | 108   |
| Flare       | 13  | 1066  |
| House-votes | 17  | 435   |

Table 2: The running time of our algorithm on the benchmark datasets. We measure in seconds the time spent on the dynamic programming phase (DP) as well as sampling of  $10^5$ ,  $10^6$ , and  $10^7$  graphs. These include the time spent finding the number of junction trees per sample.

| Dataset     | DP    | $10^5$ | $10^6$ | $10^7$ |
|-------------|-------|--------|--------|--------|
| Asia        | 0.018 | 0.32   | 3.2    | 33     |
| Bridges     | 5.6   | 0.66   | 5.5    | 54     |
| Flare       | 24    | 1.5    | 8.8    | 77     |
| House-votes | 7497  | 16     | 40     | 167    |

trees, a method described by Thomas and Green (2009) has, to our knowledge, the best worst case guarantees, running in time  $O(n^2)$ . It also starts by finding a single junction tree of the graph, which in our case is already given.

## 6 EXPERIMENTS

We report experimental results on the proposed sampling and estimation methods, using benchmark datasets (Table 1). *Asia* is sampled from a network defined by Lauritzen and Spiegelhalter (1988) and others are from the UCI repository (Bache and Lichman 2013). For all datasets we use the Dirichlet–multinomial model with the equivalent sample size parameter 1 (Dawid and Lauritzen 1993, Heckerman et al. 1995). Our C++ implementation<sup>1</sup> uses the straightforward  $O(4^n)$  time dynamic programming (Sect. 2) and the adaptive variant of the sampling phase (Sect. 4). The running times are detailed in Table 2.

### 6.1 PREDICTION

We first study how well averaging over graphs using our sampling method performs on a prediction task, as compared to using a single maximum-a-posteriori graph. Specifically, we split each dataset into a fixed test set and a training set. From the training data we then learn both a maximum-a-posteriori graph and the full posterior given a uniform prior over decomposable graphs and measure how the probability of the test data given the training data behaves under these models as the size of the training set varies.

The maximum-a-posteriori graph is obtained using the

<sup>1</sup>Our implementation is available at [www.cs.helsinki.fi/u/jwkangas/junctor](http://www.cs.helsinki.fi/u/jwkangas/junctor).

method of Kangas et al. (2014). The full posterior model is approximated by sampling  $2^n$  graphs, where  $n$  is the number of variables in the dataset, and then measuring the average probability of the test data over the sampled graphs (weighting each graph by the number of its RPTs).

Intuitively, the full posterior should in general yield better results, as the maximum-a-posteriori graph has a tendency to overfit the training data, especially for a small training set. Our results (Fig. 1) conform to this expectation. We observe that the probability of the test data under the full posterior model tends to be significantly higher for small training sets and the gap diminishes as the size of the training set grows.

### 6.2 EDGE POSTERIOR PROBABILITIES

In the second set of experiments we apply the Monte Carlo methods described in Sect. 5 to estimating the posterior probabilities of individual edges under the uniform prior over decomposable models.

For each dataset, we use the method from Sect. 5.2 to estimate the posterior probability of each edge. We compute the *estimation error*, i.e., the difference between an edge’s estimated probability and our best available estimate for the probability. Ideally, we would like the best available estimate to be the true posterior probability of the edge. For *Asia* we are able to obtain the true probability by enumerating all decomposable graphs up to 8 nodes. As this task comes infeasible for larger  $n$ , for other datasets we instead compare to the best estimate given by our method after drawing one million samples. Such “self-comparison” is still useful for studying the rate of convergence.

To quantify convergence, we measure how the maximum estimation error over edges develops as we draw more samples (Fig. 2). We also study the distribution of the error in a more refined way by counting the number of edges whose error exceeds  $10^{-k}$  for  $k \in \{1, \dots, 4\}$  (Fig. 3). We observe that the rate of convergence remains steady for each dataset. In all cases we reach a maximum error of about 0.01 or less after  $10^5$  samples and a lot sooner for smaller datasets.

## 7 CONCLUDING REMARKS

We have presented algorithms for Bayesian learning of moderate-size decomposable graphical models. Unlike the recent related algorithm that finds a single optimal model (Kangas et al. 2014), our algorithm enables more principled treatment of the posterior uncertainty. Unlike the brute-force approach, our algorithm scales well beyond 8 variables, up to about 20 variables. Unlike the popular MCMC methods, our algorithm enjoys accuracy guarantees (see also below). We believe our algorithms can be valuable for solving actual data-analysis instances, for testing the performance of other (approximate) methods, and as a building block for developing new methods that scale to larger instances.



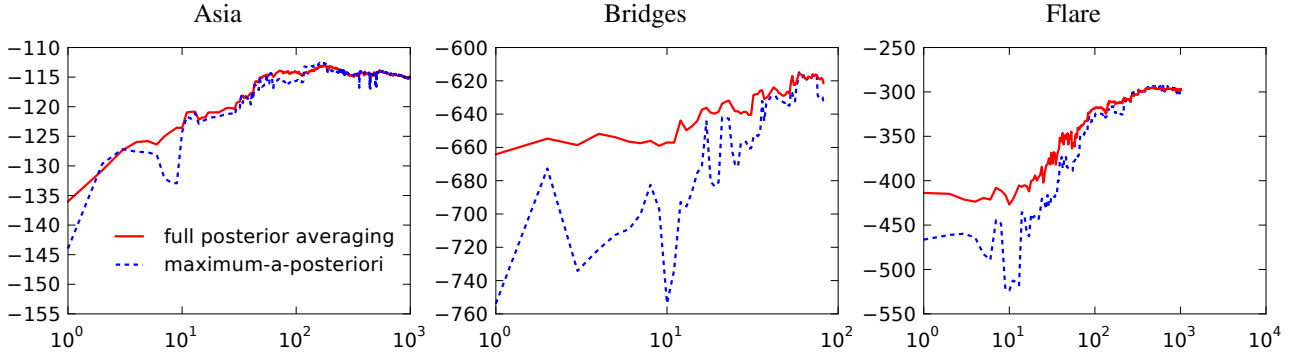


Figure 1: The (log unnormalized) probability of the test data given the training data (y-axis) as an average under sampled graphs and under the maximum-a-posteriori model for various sizes of the training set (x-axis).

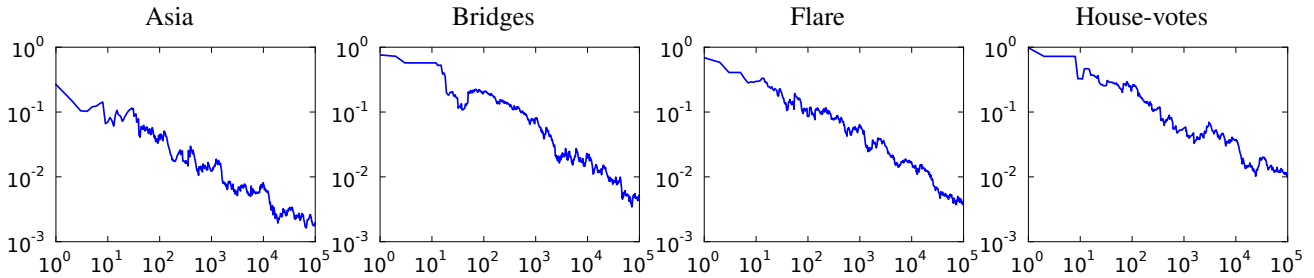


Figure 2: The maximum error of the estimate over all edges (y-axis) as the number of samples (x-axis) grows.

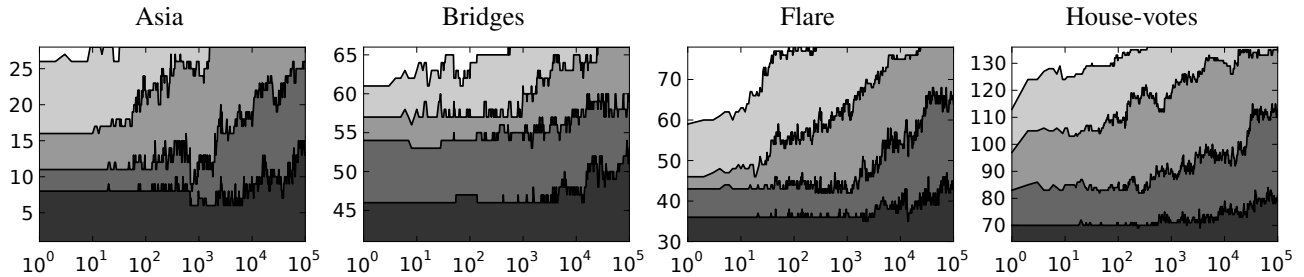


Figure 3: The number of edges (y-axis) with an estimation error less than  $10^{-k}$  as the number of samples (x-axis) grows. The curves represents  $k = 1, \dots, 4$  from top to down.

While this work has touched several aspects of accurate Bayesian learning of decomposable graphs, it also leaves many questions for future work. Some are rather straightforward extension and implementation issues: From a practical point of view, the most important one is to make the algorithms accommodate a user-specified upper bound on the clique sizes, and thereby expedite computations and reduce memory requirements. Another issue is to tune the recursive sampling schemes so as to fully exploit the (typical) low entropy of the distributions. Conceptually, the biggest shortcoming in our current importance sampling implementation is the lack of controllable approximation guarantees. However, we believe this gap can be closed in an efficient and practical way by using the so-called optimal Monte

Carlo algorithms (Cheng 2001, Dagum et al. 2000), as the sampled graphs tend to have relatively few junction trees.

The main open research questions are: Can the asymptotically faster algorithm be implemented to run fast also in practice? Are there significantly faster and, particularly, more space-efficient algorithms for Bayesian learning of decomposable graphs, with good accuracy guarantees?

### Acknowledgements

The authors thank the anonymous reviewers for valuable suggestions to improve the presentation. This work was supported in part by the Academy of Finland, Grant 276864 “Supple Exponential Algorithms” (M.K.).

## References

- H. Abel and A. Thomas. Accuracy and computational efficiency of a graphical modeling approach to linkage disequilibrium estimation. *Statistical Applications in Genetics and Molecular Biology*, 10:1–15, 2011.
- K. Bache and M. Lichman. UCI machine learning repository, 2013.
- A. Björklund, T. Husfeldt, P. Kaski, and M. Koivisto. Fourier meets Möbius: fast subset convolution. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing (STOC)*, pages 67–74. ACM, 2007.
- A. Björklund, M. Koivisto, T. Husfeldt, J. Nederlof, P. Kaski, and P. Parviainen. Fast zeta transforms for lattices with few irreducibles. In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1436–1444. SIAM, 2012.
- J. Cheng. Sampling algorithms for estimating the mean of bounded random variables. *Computational Statistics*, 16:1–23, 2001.
- J. Corander, M. Gyllenberg, and T. Koski. Bayesian model learning based on a parallel MCMC strategy. *Statistics and Computing*, 16(4):355–362, 2006.
- J. Corander, T. Janhunen, J. Rintanen, H. Nyman, and J. Pensar. Learning chordal Markov networks by constraint satisfaction. In *Advances in Neural Information Processing Systems 26 (NIPS)*, pages 1349–1357. Curran Associates, Inc., 2013.
- P. Dagum, R. M. Karp, M. Luby, and S. M. Ross. An optimal algorithm for Monte Carlo estimation. *SIAM Journal on Computing*, 29(5):1484–1496, 2000.
- A. P. Dawid and S. L. Lauritzen. Hyper Markov laws in the statistical analysis of decomposable graphical models. *The Annals of Statistics*, 21(3):1272–1317, 1993.
- P. Giudici and P. J. Green. Decomposable graphical Gaussian model determination. *Biometrika*, 86(4):785–801, 1999.
- P. J. Green and A. Thomas. Sampling decomposable graphs using a Markov chain on junction trees. *Biometrika*, 100(1):91–110, 2013.
- D. Heckerman, D. Geiger, and D. M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20:197–243, 1995.
- K. Kangas, M. Koivisto, and T. Niinimäki. Learning chordal Markov networks by dynamic programming. In *Advances in Neural Information Processing Systems 27 (NIPS)*, pages 2357–2365. Curran Associates, Inc., 2014.
- R. Kennes and P. Smets. Computational aspects of the Möbius transformation. In *Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 401–416. Elsevier, 1990.
- S. L. Lauritzen. *Graphical Models*. Oxford University Press, 1996.
- S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, 50(2):157–224, 1988.
- D. Madigan and J. York. Bayesian graphical models for discrete data. *International Statistical Review*, 63:215–232, 1995.
- N. Srebro. Maximum likelihood bounded tree-width Markov networks. *Artificial Intelligence*, 143(1):123–138, 2003.
- C. Tarantola. MCMC model determination for discrete graphical models. *Statistical Modelling*, 4(1):39–61, 2004.
- A. Thomas and P. J. Green. Enumerating the junction trees of a decomposable graph. *Journal of Computational and Graphical Statistics*, 18:930–940, 2009.
- M. Vose. A linear algorithm for generating random numbers with a given distribution. *IEEE Transactions on Software Engineering*, 17:972–975, 1991.
- F. Yates. *The design and analysis of factorial experiments*. Imperial Bureau of Soil Science. Harpenden, 1937.

---

# Novel Bernstein-like Concentration Inequalities for the Missing Mass

---

**Bahman Yari Saeed Khanloo**  
Monash University  
bahman.khanloo@monash.edu

**Gholamreza Haffari**  
Monash University  
gholamreza.haffari@monash.edu

## Abstract

We are concerned with obtaining novel concentration inequalities for the *missing mass*, i.e. the total probability mass of the outcomes not observed in the sample. We not only derive - for the first time - distribution-free Bernstein-like deviation bounds with *sublinear* exponents in deviation size for missing mass, but also improve the results of McAllester and Ortiz (2003) and Berend and Kontorovich (2013, 2012) for small deviations which is the most interesting case in learning theory. It is known that the majority of standard inequalities cannot be directly used to analyze heterogeneous sums i.e. sums whose terms have large difference in magnitude. Our generic and intuitive approach shows that the heterogeneity issue introduced in McAllester and Ortiz (2003) is resolvable at least in the case of missing mass via regulating the terms using our novel thresholding technique.

## 1 INTRODUCTION

Missing mass is the total probability associated to the outcomes that have not been seen in the sample which is one of the important quantities in machine learning and statistics. It connects density estimates obtained from a given sample to the population for discrete distributions: the less the missing mass, the more useful the information that can be extracted from the dataset. Roughly speaking, the more the missing mass is the less we can discover about the true unknown underlying distribution which would imply the less we can statistically generalize to the whole population. In other words, missing mass measures how representative a given dataset is assuming that it has been sampled according to the true distribution.

Often, one is interested in understanding the behaviour of the missing mass as a random variable. One of the

important approaches in such studies involves bounding the fluctuations of the random variable around a certain quantity namely its mean. Concentration inequalities are powerful tools for performing analysis of this type. Let  $X$  be any non-negative real-valued random variable with finite mean. The goal is to establish for any  $\epsilon > 0$ , probability bounds of the form

$$\begin{aligned}\mathbb{P}(X - \mathbb{E}[X] \leq -\epsilon) &\leq \exp(-\eta_l(\epsilon)), \\ \mathbb{P}(X - \mathbb{E}[X] \geq \epsilon) &\leq \exp(-\eta_u(\epsilon)),\end{aligned}\quad (1)$$

where  $\eta_l(\epsilon)$  and  $\eta_u(\epsilon)$  are some non-decreasing functions of  $\epsilon$  and where it is desirable to find the largest such functions for variable  $X$  and for the ‘target’ interval of  $\epsilon$ . These bounds are commonly called lower and upper deviations bounds respectively. In most practical scenarios, we are in a non-asymptotic setting where we have access to a sample  $X_1, \dots, X_n$  and we would like to derive concentration inequalities that explicitly describe dependence on sample size  $n$ . Namely, we would like to obtain bounds of the form

$$\begin{aligned}\mathbb{P}(X - \mathbb{E}[X] \leq -\epsilon) &\leq \exp(-\eta_l(\epsilon, n)), \\ \mathbb{P}(X - \mathbb{E}[X] \geq \epsilon) &\leq \exp(-\eta_u(\epsilon, n)),\end{aligned}\quad (2)$$

where  $\eta_l(\epsilon, n)$  and  $\eta_u(\epsilon, n)$  are both non-decreasing functions of  $\epsilon$  and  $n$ . Many of such bounds are distribution-free i.e. they hold irrespective of the underlying distribution.

McAllester and Schapire (2000) established concentration inequalities for the missing mass for the first time. A follow-up work by McAllester and Ortiz (2003) pointed out inadequacy of standard inequalities, developed a thermodynamical viewpoint for addressing this issue and sharpened these bounds. Berend and Kontorovich (2013) further refined the bounds via arguments similar to Kearns-Saul inequality (Kearns and Saul (1998)) and logarithmic Sobolev inequality (Boucheron et al. (2013)). These previous works, however, not only involve overly specific approaches to concentration and handling heterogeneity issue but also do not yield sharp bounds for small deviations which is the most interesting case in learning theory.

In this paper, we shall derive distribution-free concentration inequalities for missing mass in a novel way. The

primary objective of our approach is to introduce a notion of *heterogeneity control* which allows us to *regulate* the magnitude of bins in histogram of the discrete distribution being analyzed. This mechanism in turn enables us to control the behaviour of central quantities such as the variance or martingale differences of the random variable in question. These are the main quantities that appear in standard concentration inequalities such as Bernstein, Bennett and McDiarmid just to name a few. Consequently, instead of discovering a new method for bounding fluctuations of each random variable of interest, we will be able to directly apply standard inequalities to obtain probabilistic bounds on many discrete random variables including missing mass.

The rest of the paper is structured as follows. Section 2 contains the background information and introduces the notations. Section 3 outlines motivations and the main contributions. In Section 4, we explain negative dependence, information monotonicity and develop a few fundamental tools whereas Section 5 presents the proofs of our upper and lower deviation bounds based on these tools. Finally, Section 6 concludes the paper and compares our bounds with existing results for small deviations.

## 2 PRELIMINARIES

In this section, we will provide definitions, notations and other background material.

Consider  $P : \mathcal{I} \rightarrow [0, 1]$  to be a fixed but unknown discrete distribution on some finite or countable non-empty set  $\mathcal{I}$  with  $|\mathcal{I}| = N$ . Let  $\{w_i : i \in \mathcal{I}\}$  be the probability (or frequency) of drawing the  $i$ -th outcome. Moreover, suppose that we observe an i.i.d. sample  $\{X_j\}_{j=1}^n$  from this distribution with  $n$  being the sample size. Now, missing mass is defined as the total probability mass corresponding to the outcomes that are not present in our sample. Namely, missing mass is a random variable that can be expressed as:

$$Y := \sum_{i \in \mathcal{I}} w_i Y_i, \quad (3)$$

where we define each  $\{Y_i : i \in \mathcal{I}\}$  to be a Bernoulli variable that takes on 0 if the  $i$ -th outcome exists in the sample and 1 otherwise. Namely, we have

$$Y_i = \mathbb{1}_{[(X_1 \neq i) \wedge (X_2 \neq i) \wedge \dots \wedge (X_n \neq i)]}. \quad (4)$$

We assume that for all  $i \in \mathcal{I}$ ,  $w_i > 0$  and  $\sum_{i \in \mathcal{I}} w_i = 1$ . Denote  $P(Y_i = 1) = q_i$  and  $P(Y_i = 0) = 1 - q_i$  and let us suppose that  $Y_i$ s are independent: as we will see later in this section, such an assumption will not impose a burden on our proof structure and flow. Hence, we will have that  $q_i = \mathbb{E}[Y_i] = (1 - w_i)^n \leq e^{-nw_i}$  where  $q_i \in (0, 1)$ . Namely, defining  $f : (1, n) \rightarrow (e^{-n}, \frac{1}{e}) \subset (0, 1)$  where  $f(\theta) = e^{-\theta}$  with  $\theta \in D_f$  and taking  $w_i > \frac{\theta}{n}$  amounts to  $q_i(w_i) \leq f(\theta)$ . This provides a basis for our ‘thresholding’ technique that we will employ in our proof.

Choosing the representation (3) for missing mass, one has

$$\mathbb{E}[Y]_{\mathcal{I}} = \sum_{i \in \mathcal{I}} w_i q_i = \sum_{i \in \mathcal{I}} w_i (1 - w_i)^n, \quad (5)$$

$$V[Y]_{\mathcal{I}} = \sum_{i \in \mathcal{I}} w_i^2 \text{VAR}[Y_i], \quad (6)$$

$$\underline{\sigma}_{\mathcal{I}}^2 := \sum_{i \in \mathcal{I}} w_i \text{VAR}[Y_i], \quad (7)$$

where we have introduced the weighted variance notation  $\underline{\sigma}_{\mathcal{I}}^2$  and where each quantity is attached to a set over which it is defined. Note that  $\text{VAR}[Y_i]$  is the individual variance corresponding to  $Y_i$  which is defined as

$$\text{VAR}[Y_i] = q_i(1 - q_i) = (1 - w_i)^n(1 - (1 - w_i)^n). \quad (8)$$

One can define the above quantities not just over the set  $\mathcal{I}$  but on some (proper) subset of it that may depend on or be described by some variable(s) of interest. For instance, in our proofs the variable  $\theta$  may be responsible for choosing  $\mathcal{I}_{\theta} \subseteq \mathcal{I}$  over which the above quantities will be evaluated. For lower deviation and upper deviation, we find it convenient to refer to the associated set by  $\mathcal{L}$  and  $\mathcal{U}$  respectively. Likewise, we will use subscripts  $l$  and  $u$  to refer to objects that characterize lower deviation and upper deviation respectively. Also, we use the notation  $Y^{ij} = Y_i, \dots, Y_j$  to refer to sequence of variables whose index starts at  $i$ -th variable and ends at  $j$ -th variable. Finally, other notation or definitions may be introduced within the body of the proof when required.

We will encounter Lambert  $W$ -function - also known as product logarithm function - in this paper which describes the inverse relation of  $f(x) = xe^x$  and which cannot be expressed in terms of elementary functions. This function is double-valued when  $x \in \mathbb{R}$ . However, it becomes invertible in restricted domain. The lower branch of it is denoted by  $W_{-1}(\cdot)$ , which is the only branch that will prove beneficial in this paper. The reader is advised to refer to Corless et al. (1996) for a detailed treatment.

Throughout the paper, we shall use the convention that capital letters refer to random variables whereas lower case letters correspond to realizations thereof.

We will utilize Bernstein’s inequality in our derivation. Suitable representations of this result are outlined below without the proof.

**Theorem.** [Bernstein] *Let  $Z_1, \dots, Z_N$  be independent zero-mean random variables such that one has  $|Z_i| \leq \alpha$  almost surely for all  $i$ . Then, using Bernstein’s inequality (Bernstein (1924)) one obtains for all  $\epsilon > 0$ :*

$$\mathbb{P}\left(\sum_{i=1}^N Z_i > \epsilon\right) \leq \exp\left(-\frac{\epsilon^2}{2(V + \frac{1}{3}\alpha\epsilon)}\right), \quad (9)$$

where  $V = \sum_{i=1}^N \mathbb{E}[Z_i^2]$ .

Now, consider the sample mean  $\bar{Z} = n^{-1} \sum_{i=1}^n Z_i$  and let  $\bar{\sigma}^2$  be the sample variance, namely  $\bar{\sigma}^2 := n^{-1} \sum_{i=1}^n \text{VAR}[Z_i] = n^{-1} \sum_{i=1}^n \mathbb{E}[Z_i^2]$ . So, using (9) with  $n \cdot \epsilon$  in the role of  $\epsilon$ , we get

$$\mathbb{P}(\bar{Z} > \epsilon) \leq \exp\left(-\frac{n\epsilon^2}{2(\bar{\sigma}^2 + \frac{1}{3}\alpha\epsilon)}\right). \quad (10)$$

If  $Z_1, \dots, Z_n$  are, moreover, not just independent but also identically distributed, then  $\bar{\sigma}^2$  is equal to  $\sigma^2$  i.e. the variance of each  $Z_i$ . The latter presentation makes explicit: (1) the exponential decay with  $n$ ; (2) the fact that for  $\bar{\sigma}^2 \leq \epsilon$  we get a tail probability with exponent of order  $n\epsilon$  rather than  $n\epsilon^2$  (Lugosi (2003); Boucheron et al. (2013)) which has the potential to yield stronger bounds for small  $\epsilon$ .

### 3 MOTIVATIONS AND MAIN RESULTS

In this section, we motivate this work by pointing out the heterogeneity challenge and how we approach it. Our bounds also improve the functional form of the exponent, which is of independent significance. In the final part of this section, we summarize our main results.

#### 3.1 The Challenge and the Remedy

McAllester and Ortiz (2003) point out that for highly heterogeneous sums of the form (3), the standard form of Bernstein's inequality (9) does not lead to concentration inequalities of form (10): at least for the upper deviation of the missing mass, (9) does not imply any non-trivial bounds of the form (2). The reason is basically the fact that the  $w_i$  can vary wildly: some can be of order  $O(1/n)$ , other may be constants independent of  $n$ . For similar reasons, other standard inequalities such as Bennett, Angluin-Valiant and Hoeffding cannot be used to get bounds on the missing mass of the form (2) either (McAllester and Ortiz (2003)).

Having pointed out the deficiency of these standard inequalities, McAllester and Ortiz (2003) succeed in giving bounds of the form (2) on the missing mass, for a function  $\eta(\epsilon, n) \propto n\epsilon^2$ , both with a direct argument and using the Kearns-Saul inequality (Kearns and Saul (1998)). Recently, the constants appearing in the bounds were refined by Berend and Kontorovich (2013). The bounds proven by McAllester and Ortiz (2003) and Berend and Kontorovich (2013) are qualitatively similar to Hoeffding bounds for i.i.d. random variables: they do *not* improve the functional form from  $n\epsilon^2$  to  $n\epsilon$  for small variances.

This leaves open the question whether it is also possible to derive bounds which are more reminiscent of the Bernstein bound for i.i.d. random variables (10) which does exploit variance. In this paper, we show that the answer is a qualified yes: we give bounds that depend on weighted variance  $\underline{\sigma}^2$  defined in (7) rather than sample variance  $\bar{\sigma}^2$  as in (10) which is tight exactly in the

important case when  $\underline{\sigma}^2$  is small, and in which the denominator in (10) is specified by a factor depending on  $\epsilon$ ; in the special case of the missing mass, this factor turns out to be logarithmic in  $\epsilon$  and a free parameter  $\gamma$  as it will become clear later.

We derive - using Bernstein's inequality - novel bounds on missing mass that take into account explicit variance information with more accurate scaling and demonstrate their superiority for small deviations.

#### 3.2 Main Results

Consider the following functions

$$\gamma_\epsilon = -2W_{-1}\left(-\frac{\epsilon}{2\sqrt{e}}\right), \quad (11)$$

$$c(\epsilon) = \frac{3(\gamma_\epsilon - 1)}{5\gamma_\epsilon^2}. \quad (12)$$

Let  $Y$  denote the missing mass,  $n$  the sample size and  $\epsilon$  the deviation size.

**Theorem 1.** *For any  $0 < \epsilon < 1$  and any  $n \geq \lceil \gamma_\epsilon \rceil - 1$ , we obtain the following upper deviation bound*

$$\mathbb{P}(Y - \mathbb{E}[Y] \geq \epsilon) \leq e^{-c(\epsilon) \cdot n\epsilon}. \quad (13)$$

**Theorem 2.** *For any  $0 < \epsilon < 1$  and any  $n \geq \lceil \gamma_\epsilon \rceil - 1$ , we obtain the following lower deviation bound*

$$\mathbb{P}(Y - \mathbb{E}[Y] \leq -\epsilon) \leq e^{-c(\epsilon) \cdot n\epsilon}. \quad (14)$$

**Corollary 1.** *For any  $0 < \epsilon < 1$  and any  $n \geq \lceil \gamma_\epsilon \rceil - 1$ , using union bound we obtain the following deviation bound*

$$\mathbb{P}(|Y - \mathbb{E}[Y]| \geq \epsilon) \leq 2e^{-c(\epsilon) \cdot n\epsilon}. \quad (15)$$

The proof of the above theorems is provided in Section 5. However, let us develop a few tools in Section 4 which will be used later in our proofs.

### 4 NEGATIVE DEPENDENCE AND INFORMATION MONOTONICITY

Probabilistic analysis of most random variables and specifically the derivation of the majority of probabilistic bounds rely on independence assumption between variables which offers considerable simplification and convenience. Many random variables including the missing mass, however, consist of random components that are not independent.

Fortunately, even in cases where independence does not hold, one can still use some standard tools and methods provided variables are dependent in specific ways. The following notions of dependence are among the common ways that prove useful in these settings: negative association and negative regression.

#### 4.1 Negative Dependence and Chernoff's Exponential Moment Method

Our proof involves variables with a specific type of dependence known as negative association. One can infer concentration of sums of negatively associated random variables from the concentration of sums of their independent copies in certain situations. In exponential moment method, this property allows us to treat such variables as independent in the context of probability inequalities as we shall elaborate later in this section.

In the sequel, we present negative association and regression and supply tools that will be essential in proofs.

**Negative Association:** Any real-valued random variables  $X_1$  and  $X_2$  are negatively associated if

$$\mathbb{E}[X_1 X_2] \leq \mathbb{E}[X_1] \cdot \mathbb{E}[X_2]. \quad (16)$$

More generally, a set of random variables  $X_1, \dots, X_m$  are negatively associated if for any disjoint subsets  $A$  and  $B$  of the index set  $\{1, \dots, m\}$ , we have

$$\mathbb{E}[X_i X_j] \leq \mathbb{E}[X_i] \cdot \mathbb{E}[X_j] \quad \text{for } i \in A, j \in B. \quad (17)$$

**Stochastic Domination:** Assume that  $X$  and  $Y$  are real-valued random variables. Then,  $X$  is said to stochastically dominate  $Y$  if for all  $a$  in the range of  $X$  and  $Y$  we have

$$P(X \geq a) \geq P(Y \geq a). \quad (18)$$

We use the notation  $X \succeq Y$  to reflect (18) in short.

**Stochastic Monotonicity:** A random variable  $Y$  is stochastically non-decreasing in random variable  $X$  if

$$x_1 \leq x_2 \implies P(Y|X = x_1) \leq P(Y|X = x_2). \quad (19)$$

Similarly,  $Y$  is stochastically non-increasing in  $X$  if

$$x_1 \leq x_2 \implies P(Y|X = x_1) \geq P(Y|X = x_2). \quad (20)$$

The notations  $(Y|X = x_1) \preceq (Y|X = x_2)$  and  $(Y|X = x_1) \succeq (Y|X = x_2)$  represent the above definitions using the notion of stochastic domination. Also, we will use shorthands  $Y \uparrow X$  and  $Y \downarrow X$  to refer to the relations described by (19) and (20) respectively.

**Negative Regression:** Random variables  $X$  and  $Y$  have negative regression dependence relation if  $X \downarrow Y$ .

Dubhashi and Ranjan (1998) as well as Joag-Dev and Proschan (1983) summarize numerous notable properties of negative association and negative regression. Specifically, the former provides a proposition that indicates that Hoeffding-Chernoff bounds apply to sums of negatively associated random variables. Further, McAllester and Ortiz (2003) generalize these observations to essentially any concentration result derived based on the exponential moment

method by drawing a connection between deviation probability of a discrete random variable and Chernoff's entropy of a related distribution.

We provide a self-standing account by presenting the proof for some of these existing results as well as developing several generic tools that are applicable beyond missing mass problem.

**Lemma 1. [Binary Stochastic Monotonicity]** Let  $Y$  be a binary random variable (Bernoulli) and let  $X$  take on values in a totally ordered set  $\mathcal{X}$ . Then, one has

$$Y \downarrow X \implies X \downarrow Y. \quad (21)$$

*Proof.* For any  $x$ , we have

$$\begin{aligned} P(Y = 1 | X \leq x) &\geq \inf_{a \leq x} P(Y = 1 | X = a) \\ &\geq \sup_{a > x} P(Y = 1 | X = a) \\ &\geq P(Y = 1 | X > x). \end{aligned} \quad (22)$$

The above argument implies that random variables  $Y$  and  $\mathbf{1}_{X > x}$  are negatively associated and since the expression  $P(X > x | Y = 1) \leq P(X > x | Y = 0)$  holds for all  $x \in \mathcal{X}$ , it follows that  $X \downarrow Y$ .  $\square$

**Lemma 2. [Independent Binary Negative Regression]**

Let  $X_1, \dots, X_m$  be negatively associated random variables and  $Y_1, \dots, Y_m$  be binary random variables (Bernoulli) such that either  $Y_i \downarrow X_i$  or  $Y_i \uparrow X_i$  holds for all  $i \in \{1, \dots, m\}$ . Then  $Y_1, \dots, Y_m$  are negatively associated.

*Proof.* For any disjoint subsets  $A$  and  $B$  of  $\{1, \dots, m\}$ , taking  $i \in A$  and  $j \in B$  we have

$$\mathbb{E}[Y_i Y_j] = \mathbb{E}[\mathbb{E}[Y_i Y_j | X_1, \dots, X_m]] \quad (23)$$

$$= \mathbb{E}[\mathbb{E}[Y_i | X_i] \cdot \mathbb{E}[Y_j | X_j]] \quad (24)$$

$$\leq \mathbb{E}[\mathbb{E}[Y_i | X_i]] \cdot \mathbb{E}[\mathbb{E}[Y_j | X_j]] \quad (25)$$

$$= \mathbb{E}[Y_i] \cdot \mathbb{E}[Y_j]. \quad (26)$$

Here, (24) holds since each  $Y_i$  only depends on  $X_i$ . Inequality (25) follows because  $X_i$  and  $X_j$  are negatively associated and we have  $\mathbb{E}[Y_i | X_i] = P(Y_i | X_i)$ .  $\square$

**Lemma 3. [Chernoff]** For any real-valued random variable  $X$  with finite mean  $\mathbb{E}[X]$  and for any  $x > 0$ , we have:

$$DP(X, x) \leq \exp(-S(X, x)), \quad (27)$$

$$S(X, x) = \sup_{\lambda} \{\lambda x - \ln(Z(X, \lambda))\}, \quad (28)$$

$$Z(X, \lambda) = \mathbb{E}[e^{\lambda X}]. \quad (29)$$

The lemma follows from the observation that for  $\lambda \geq 0$ , we have the following

$$P(X \geq x) = P(e^{\lambda X} \geq e^{\lambda x}) \leq \inf_{\lambda} \frac{\mathbb{E}[e^{\lambda X}]}{e^{\lambda x}}. \quad (30)$$

This approach is known as *exponential moment method* (Chernoff (1952)) because of the inequality in (30).

**Lemma 4. [Negative Association]** In the exponential moment method, concentration of sums of negatively associated random variables can be deduced from the concentration of sums of their independent copies.

*Proof.* Let  $X_1, \dots, X_m$  be any set of negatively associated variables. Let  $X'_1, \dots, X'_m$  be independent shadow variables, i.e., independent variables such that each  $X'_i$  is distributed identically to  $X_i$ . Let  $X = \sum_i^m X_i$  and  $X' = \sum_i^m X'_i$ . For any set of negatively associated random variables, one has  $S(X, \epsilon) \geq S(X', \epsilon)$  since:

$$\begin{aligned} Z(X, \lambda) &= \mathbb{E}[e^{\lambda X}] = \mathbb{E}\left[\prod_i^m e^{\lambda X_i}\right] \\ &\leq \prod_i^m \mathbb{E}[e^{\lambda X_i}] = \mathbb{E}[e^{\lambda X'}] = Z(X', \lambda). \end{aligned} \quad (31)$$

The lemma is due to McAllester and Ortiz (2003) which follows from definition of entropy function  $S$  given by (28).  $\square$

This lemma is very helpful in the context of large deviation bounds: it implies that one can treat negatively associated variables as if they were independent (McAllester and Ortiz (2003); Dubhashi and Ranjan (1998)).

**Lemma 5. [Balls and Bins]** Let  $S$  be any sample comprising  $n$  items drawn i.i.d. from a fixed distribution on integers  $\mathcal{N} = \{1, \dots, N\}$  (bins). Define  $C_i$  to be the number of times that integer  $i$  occurs in  $S$ . The random variables  $C_1, \dots, C_N$  are negatively associated.

*Proof.* Let  $f$  and  $g$  be non-decreasing and non-increasing functions respectively. We have

$$(f(x) - f(y))(g(x) - g(y)) \leq 0. \quad (32)$$

Further, assume that  $X$  is a real-valued random variable and  $Y$  is an independent shadow variable corresponding to  $X$ . Exploiting (32), we obtain

$$\mathbb{E}[f(X)g(X)] \leq \mathbb{E}[f(X)] \cdot \mathbb{E}[g(X)], \quad (33)$$

which implies that  $f(X)$  and  $g(X)$  are negatively associated. Inequality (33) is an instance of Chebychev's fundamental *association inequality*.

Now, suppose without loss of generality that  $N = 2$ . Take  $X \in [0, n]$ , and consider the following functions

$$\begin{cases} f(X) = X, \\ g(X) = n - X, \end{cases} \quad (34)$$

where  $n = C_i + C_j$  is the total counts. Since  $f$  and  $g$  are non-decreasing and non-increasing functions of  $X$ , choosing  $X = f(C_i) = C_i$  we have for all  $i, j \in \mathcal{N}$  that

$$\mathbb{E}[C_i \cdot C_j] \leq \mathbb{E}[C_i] \cdot \mathbb{E}[C_j], \quad (35)$$

which concludes the proof for  $N = 2$ . Now, taking  $f(C_i) = C_i$  and  $g(C_i) = n - \sum_{j \neq i} C_j$  where  $n = \sum_{k=1}^N C_k$ , for  $N > 2$  the same argument implies that  $C_i$  and  $C_j$  are negatively associated for all  $i \in \mathcal{N}$  and  $j \in \mathcal{N} \setminus i$ . That is to say, any increase in  $C_i$  will cause a decrease in some or all of  $C_j$  variables with  $j \neq i$  and vice versa. It is easy to verify that the same is true for any disjoint subsets of the set  $\{C_1, \dots, C_N\}$ .  $\square$

**Lemma 6. [Monotonicity]** For any negatively associated random variables  $X_1, \dots, X_m$  and any non-decreasing functions  $f_1, \dots, f_m$ , we have that  $f_1(X_1), \dots, f_m(X_m)$  are negatively associated. The same holds if the functions  $f_1, \dots, f_m$  were non-increasing.

**Remark:** The proof is in the same spirit as that of association inequality (33) and motivated by composition rules for monotonic functions that one can repeatedly apply to (32).

**Lemma 7. [Union]** The union of independent sets of negatively associated random variables yields a set of negatively associated random variables.

Suppose that  $X$  and  $Y$  are independent vectors each of which comprising a negatively associated set. Then, the concatenated vector  $[X, Y]$  is negatively associated.

*Proof.* Let  $[X_1, X_2]$  and  $[Y_1, Y_2]$  be some arbitrary partitions of  $X$  and  $Y$  respectively and assume that  $f$  and  $g$  are non-decreasing functions. Then, one has

$$\begin{aligned} \mathbb{E}[f(X_1, Y_1) \cdot g(X_2, Y_2)] &= \\ \mathbb{E}[\mathbb{E}[f(X_1, Y_1) \cdot g(X_2, Y_2) \mid Y_1, Y_2]] &\leq \\ \mathbb{E}[\mathbb{E}[f(X_1, Y_1) \mid Y_1] \cdot \mathbb{E}[g(X_2, Y_2) \mid Y_2]] &\leq \\ \mathbb{E}[\mathbb{E}[f(X_1, Y_1) \mid Y_1]] \cdot \mathbb{E}[\mathbb{E}[g(X_2, Y_2) \mid Y_2]] &= \\ \mathbb{E}[f(X_1, Y_1)] \cdot \mathbb{E}[g(X_2, Y_2)]. \end{aligned} \quad (36)$$

The first inequality is due to independence of  $[X_1, X_2]$  from  $[Y_1, Y_2]$  which results in negative association being preserved under conditioning and the second inequality follows because  $[Y_1, Y_2]$  are negatively associated (Joag-Dev and Proschan (1983)). The same holds if  $f$  and  $g$  were non-increasing functions.  $\square$

**Lemma 8. [Splitting]** Splitting an arbitrary subset of bins of any fixed discrete distribution yields a set of negatively associated random bins.

*Proof.* Let  $w = (w_1, \dots, w_m)$  be a discrete distribution and  $\mathcal{W} = \{W_1, \dots, W_m\}$  be the associated set of random bins. Assume that  $w_i$  is split into  $k$  bins  $\mathcal{W}_i^S = \{W_{i1}, \dots, W_{ik}\}$  such that  $w_i = \sum_{j=1}^k W_{ij}$ . Then, by Lemma 5 members of split set  $\mathcal{W}_i^S$  are negatively associated. Clearly, the same holds for all  $1 \leq i \leq m$  as well as any other subset of set  $\mathcal{W}$ . Moreover, for all  $1 \leq i \leq m$  the sets  $\mathcal{W}_i^S$  and  $\mathcal{W} \setminus W_i$  are negatively associated by Lemma 5 and Lemma 7.  $\square$

**Lemma 9. [Absorption]** Absorbing any subset of bins of a discrete distribution yields negatively associated bins.

*Proof.* Let  $w = (w_1, \dots, w_N)$  be a discrete distribution and let  $\mathcal{W} = \{W_1, \dots, W_N\}$  be the associated set of random bins. Assume without loss of generality that  $\mathcal{W}^A = \{W_1^A, \dots, W_{N-1}^A\}$  is the absorption-induced set of random bins where  $w_N$  is absorbed to produce  $w^A = (w_1^A, \dots, w_{N-1}^A)$  and where  $w_i^A = w_i + \frac{w_N}{N-1}$  for  $i = 1, \dots, N-1$ . So,  $w_N$  is discarded and we have  $\sum_{i=1}^{N-1} W_i^A = 1 - w_N$ . The rest of the proof concerns applying Lemma 5 to the absorb set  $\mathcal{W}^A$ . The same holds if we absorb  $w_N$  to a subset of  $\mathcal{W} \setminus W_N$ .  $\square$

## 4.2 Negative Dependence and the Missing Mass

For missing mass, the variables  $W_i = \frac{C_i}{n}$  are negatively associated owing to Lemma 5 and linearity of expectation. Also, one has  $\forall i : Y_i \downarrow W_i$ . So, by Lemma 1 we infer that  $\forall i : W_i \downarrow Y_i$ . Now,  $Y_1, \dots, Y_N$  are negatively associated because they are a set of independent binary variables with negative regression dependence (Lemma 2). Thus, concentration variables  $Z_i = w_i Y_i - \mathbb{E}[w_i Y_i] := \zeta(Y_i)$  are negatively associated by Lemma 6 since we have

$$\zeta(Y_i) = \begin{cases} -w_i q_i & \text{if } Y_i = 0, \\ w_i(1 - q_i) & \text{if } Y_i = 1. \end{cases} \quad (37)$$

For all  $i$ ,  $\zeta$  is a non-decreasing function of  $Y_i$ . Likewise, concentration variables  $-Z_i$  are negatively associated.

## 4.3 Information Monotonicity and Partitioning

**Lemma 10. [Information Monotonicity]** Let  $p = (p_1, \dots, p_N)$  be a discrete distribution on  $X = (x_1, \dots, x_N)$  such that for  $1 \leq i \leq N$  we have  $P(X = x_i) = p_i$ . Suppose we partition  $X$  into  $m \leq N$  non-empty disjoint groups  $G_1, \dots, G_m$ , namely

$$\begin{aligned} X &= \cup G_i, \\ \forall i \neq j : G_i \cap G_j &= \emptyset. \end{aligned} \quad (38)$$

This is called *coarse binning* since it generates a new distribution with groups  $G_i$  whose dimensionality is less than that of the original distribution. Note that once the distribution is transformed, considering any outcome  $x_i$  from the original distribution we will only have access to its group membership information; for instance, we can observe that it belongs to  $G_j$  but we will not be able to recover  $p_i$ .

Let us denote the induced distribution over the partition  $G = (G_1, \dots, G_m)$  by  $p^G = (p_1^G, \dots, p_m^G)$ . Clearly, we have

$$p_i^G = P(G_i) = \sum_{j \in G_i} P(x_j). \quad (39)$$

Now, consider the  $f$ -divergence  $D_f(p^G || q^G)$  between induced probability distributions  $p^G$  and  $q^G$ . Information

monotonicity states that information is lost as we partition elements of  $p$  and  $q$  into groups to produce  $p^G$  and  $q^G$  respectively. Namely, for any  $f$ -divergence one has

$$D_f(p^G || q^G) \leq D_f(p || q), \quad (40)$$

which is due to Csiszár (Csiszár (1977, 2008); Amari (2009)). This inequality is tight if and only if for any outcome  $x_i$  and partition  $G_j$ , we have  $p(x_i | G_j) = q(x_i | G_j)$ .

**Lemma 11. [Partitioning]** In the exponential moment method, one can establish a deviation bound for any discrete random variable  $X$  by invoking Chernoff's method on the associated discrete partition random variable  $X^G$ .

Formally, assume  $X$  and  $X_\lambda$  are discrete random variables defined on the set  $\mathcal{X}$  endowed with probability distributions  $p$  and  $p_\lambda$  respectively. Further, suppose that  $X^G$  and  $X_\lambda^G$  are discrete variables on a partition set  $\mathcal{X}^G$  endowed with  $p^G$  and  $p_\lambda^G$  that are obtained from  $p$  and  $p_\lambda$  by partitioning using some partition  $G$ . Then, we have

$$\forall x > 0 : DP(X, x) \leq \exp(-S(X^G, x)). \quad (41)$$

*Proof.* Let  $\lambda(x)$  be the optimal  $\lambda$  in (28). Then, we have

$$\begin{aligned} S(X, x) &= x\lambda(x) - \ln(Z(X, \lambda(x))) \\ &= D_{KL}(p_{\lambda(x)} || p) \\ &\geq D_{KL}(p_{\lambda(x)}^G || p^G) \\ &= S(X^G, x), \end{aligned} \quad (42)$$

where we have introduced the  $\lambda$ -induced distribution

$$P_\lambda(X = x) = \frac{e^{\lambda x}}{Z(X, \lambda)} P(X = x). \quad (43)$$

The inequality step in (42) follows from (40) and the observation that  $D_{KL}$  is an instance of  $f$ -divergence where  $f(v) = v \ln(v)$  with  $v \geq 0$ .  $\square$

## 5 PROOF OF THE MAIN RESULTS

The central idea of the proof is to regulate the terms in the sum given by (3) via controlling the magnitude of bins of the distribution using operations that preserve negative association. This mechanism will help defeat the heterogeneity issue leading to the failure of standard probability inequalities described by McAllester and Ortiz (2003).

### 5.1 Proof of Theorem 1: Upper Deviation Bound

We consider the thresholds  $\tau = \frac{\theta}{n}$  and  $\tau' = \frac{2\theta}{n}$  and reduce the problem to one in which all bins that are larger than  $\tau$  are eliminated, where  $\theta \in \mathbb{R}$  will depend on the target deviation size  $\epsilon$ .

The reduction is performed by *splitting* the bins that are larger than  $\tau$  and then *absorbing* the bins that are smaller



than  $\tau$ . This is followed by choosing a threshold that yields the sharpest bound for the choice of  $\epsilon$ . It turns out that the optimal threshold will too be a function of  $\epsilon$ .

Let  $\mathcal{I}_\tau \subseteq \mathcal{I}$  denote the subset of bins that are at most as large as  $\tau$ ,  $\mathcal{I}_\theta$  the subset of bins whose magnitude is between  $\tau$  and  $\tau'$ ,  $\mathcal{I}_{\tau'}$  the subset of bins larger than  $\tau'$  and  $\mathcal{I}'_\theta$  and  $\mathcal{I}'_{\tau'}$ , the set of bins that we obtain after splitting members of  $\mathcal{I}_\theta$  and  $\mathcal{I}_{\tau'}$ , respectively.

Now, for each  $i \in \mathcal{I} \setminus \mathcal{I}_\tau = \{\mathcal{I}_\theta \cup \mathcal{I}_{\tau'}\}$  and for some  $k \in \mathbb{N}$  that depends on  $i$  (but we suppress that notation below), we will have that  $k \cdot \tau \leq w_i < (k+1) \cdot \tau$ . For all such  $i$ , we define extra independent Bernoulli random variables  $Y_{ij}$  with  $j \in \mathcal{J}_i := \{1, \dots, k\}$  and their associated bins  $w_{ij}$ . For  $j \in \{1, \dots, k-1\}$ ,  $w_{ij} = \tau$  and  $w_{ik} = w_i - (k-1) \cdot \tau$ . In this way, all bins that are larger than  $\tau$  are split up into  $k$  bins, each of which is in-between  $\tau$  and  $\tau'$ ; more precisely, the first  $k-1$  are exactly  $\tau$  and the last one may be larger. Therefore, we consider the split random variable  $Y' = \sum_{i \in \mathcal{I}_\tau} w_i Y_i + \sum_{i \in \{\mathcal{I}'_{\tau'} \cup \mathcal{I}'_\theta\}} \sum_{j \in \mathcal{J}_i} w_{ij} Y_{ij}$  and the set  $\mathcal{U}' = \{i \mid w_i < \tau'\} = \{\mathcal{I}_\tau \cup \mathcal{I}'_\theta \cup \mathcal{I}'_{\tau'}\}$ . Furthermore, we introduce the random variable  $Y'' = \sum_{i \in \mathcal{U}''} w_i Y_i$  on the absorption-induced set  $\mathcal{U}'' = \{i \mid \tau \leq w_i < \tau'\}$ . The set  $\mathcal{U}''$  is generated from  $\mathcal{U}'$  as follows: we take the largest element  $j \in \mathcal{U}'$  with  $w_j < \tau$ , update  $w_l$  using  $w_l \leftarrow w_l + \frac{w_j}{|\mathcal{U}'|-1}$  for  $\{l \in \mathcal{U}' : l \neq j, w_l < \tau\}$  and discard  $w_j$ . Repeating this procedure gives a set of bins whose sizes are in-between  $\tau$  and  $\tau'$  plus a single bin of size smaller than  $\tau$ ; absorbing the latter into one of the members of the former with size  $\tau$  yields  $\mathcal{U}''$ .

Now, by choosing  $\theta$  such that  $f(\theta) = e^{-\theta} = \frac{\epsilon}{\gamma}$  and  $\theta = f^{-1}(\frac{\epsilon}{\gamma}) = \ln(\frac{\gamma}{\epsilon})$  for any  $0 < \epsilon < 1$  and  $e\epsilon < \gamma < e^n \epsilon$  as generic domain for  $\gamma$ , we derive the upper deviation bound for missing mass as follows

$$\mathbb{P}(Y - \mathbb{E}[Y] \geq \epsilon) \leq \quad (44)$$

$$\mathbb{P}(Y' - \mathbb{E}[Y'] \geq \epsilon) = \quad (45)$$

$$\mathbb{P}(Y' - \mathbb{E}[Y'] + (\mathbb{E}[Y'] - \mathbb{E}[Y]) \geq \epsilon) \leq \quad (46)$$

$$\mathbb{P}(Y' - \mathbb{E}[Y'] + f(\theta) \geq \epsilon) = \quad (47)$$

$$\mathbb{P}\left(Y' - \mathbb{E}[Y'] \geq \left(\frac{\gamma-1}{\gamma}\right)\epsilon\right) = \quad (48)$$

$$\exp\left(-\frac{\left(\frac{\gamma-1}{\gamma}\right)^2 \epsilon^2}{2(V_{\mathcal{U}''} + \frac{\alpha_n}{3} \cdot \left(\frac{\gamma-1}{\gamma}\right) \cdot \epsilon)}\right) \leq \quad (49)$$

$$\exp\left(-\frac{\left(\frac{\gamma-1}{\gamma}\right)^2 \epsilon^2}{2\left(\frac{\theta}{n} \cdot \epsilon + \frac{2\theta}{3n} \cdot \left(\frac{\gamma-1}{\gamma}\right) \cdot \epsilon\right)}\right) \leq \quad (50)$$

$$\inf_{\gamma} \left\{ \exp\left(-\frac{3n\epsilon(\gamma-1)^2}{10\gamma^2 \ln(\frac{\gamma}{\epsilon})}\right) \right\} = \quad (51)$$

$$e^{-c(\epsilon) \cdot n\epsilon}. \quad (52)$$

Clearly, we will have that  $\tau^* = \frac{\theta^*}{n}$  where  $\theta^* = \ln(\frac{\gamma\epsilon}{\epsilon})$ .

Inequality (45) follows because the splitting procedure cannot decrease deviation probability of missing mass.

Formally, assume without loss of generality that  $\mathcal{I} \setminus \mathcal{I}_\tau$  has only one element corresponding to  $Y_1$ ,  $\mathcal{J}_1 = \{1, 2\}$  and  $k_1 = 1$  i.e.  $w_1$  is split into two parts. Then, deviation probability of  $Y$  can be thought of as the total probability mass associated to independent Bernoulli variables  $Y_1, \dots, Y_N$  whose weighted sum is bounded below by some tail  $t > 0$ . Hence, we have

$$\begin{aligned} \mathbb{P}(Y \geq t) &= \sum_{Y^{1N}; \hat{Y} \geq t} P(Y_1, \dots, Y_N) \\ &= \sum_{Y^{1N}; \hat{Y} \geq t} R(Y_1) \cdot \prod_{i=2}^N R(Y_i) \\ &\quad + \sum_{Y^{1N}; \hat{Y} < t; Y \geq t} R(Y_1) \cdot \prod_{i=2}^N R(Y_i) \\ &= \sum_{Y^{1N}; \hat{Y} \geq t} R(Y_1) \cdot \prod_{i=2}^N R(Y_i) \\ &\quad + \sum_{Y^{1N}; \hat{Y} < t; Y \geq t, Y_1=1} R(Y_1) \cdot \prod_{i=2}^N R(Y_i) \\ &= \sum_{Y^{2N}; \hat{Y} \geq t} \prod_{i=2}^N R(Y_i) \\ &\quad + \sum_{Y^{2N}; \hat{Y} < t; Y \geq t} q_1 \cdot \prod_{i=2}^N R(Y_i), \end{aligned} \quad (53)$$

where  $\hat{Y} = \sum_{i \geq 2} w_i Y_i$  and  $R(Y_i) = q_i$  if  $Y_i = 1$  and  $R(Y_i) = 1 - q_i$  otherwise. Likewise, one can express the upper deviation probability of  $Y'$  as follows

$$\begin{aligned} \mathbb{P}(Y' \geq t) &= \sum_{Y_{1N}; \hat{Y} \geq t} R(Y_1) \cdot \prod_{i=2}^N R(Y_i) \\ &\quad + \sum_{Y_{11}, Y_{12}, Y^{2N}; \hat{Y} < t; Y' \geq t} \left(R(Y_{11}) \cdot R(Y_{12})\right) \prod_{i=2}^N R(Y_i) \\ &= \sum_{Y_{2N}; \hat{Y} \geq t} \prod_{i=2}^N R(Y_i) \\ &\quad + \sum_{Y_{11}, Y_{12}, Y^{2N}; \hat{Y} < t; Y' \geq t} \left(R(Y_{11}) \cdot R(Y_{12})\right) \prod_{i=2}^N R(Y_i) \\ &\geq \sum_{Y^{2N}; \hat{Y} \geq t} \prod_{i=2}^N R(Y_i) \\ &\quad + \sum_{Y^{2N}; \hat{Y} < t; Y' \geq t} (q_{11} \cdot q_{12}) \prod_{i=2}^N R(Y_i), \end{aligned} \quad (54)$$

where  $R(Y_{ij}) = q_{ij}$  if  $Y_{ij} = 1$  and  $R(Y_{ij}) = 1 - q_{ij}$  otherwise. Thus, combining (53) and (54) we have

$$\begin{aligned} \mathbb{P}(Y' \geq t) - \mathbb{P}(Y \geq t) &\geq \\ &\sum_{Y^{2N}; \hat{Y} < t; Y' \geq t; Y \geq t} (q_{11} \cdot q_{12} - q_1) \prod_{i=2}^N R(Y_i) = \\ &\sum_{Y^{2N}; \hat{Y} < t; Y' \geq t} (q_{11} \cdot q_{12} - q_1) \prod_{i=2}^N R(Y_i). \end{aligned} \quad (55)$$

To complete the proof for (45), we require the expression for the difference between deviation probabilities in (55) to be non-negative for all  $t > 0$  which holds if  $q_1 \leq q_{11} \cdot q_{12}$ . For the missing mass, this condition holds. Without loss of generality, assume that  $w_i$  is split into two terms; namely, we have  $w_i = w_{ij} + w_{ij'}$ . Then, we can check the above condition as follows

$$\begin{aligned} q_i &= (1 - w_i)^n \leq (1 - w_{ij})^n \cdot (1 - w_{ij'})^n \\ &= \left(1 - \underbrace{(w_{ij} + w_{ij'})}_{w_i} + \underbrace{w_{ij} \cdot w_{ij'}}_{\geq 0}\right)^n. \end{aligned} \quad (56)$$

One can verify using induction that (56) holds also for cases where the split operation produces more than two terms. Now, choosing tail size  $t = \epsilon + \mathbb{E}Y$  implies (45).

Inequality (47) follows because the gap between the expectations will be negligible. Denoting  $\mathbb{E}[Y'_i] = q'_i$ , we have

$$q'_i = \begin{cases} q_i & \text{if } i \in \mathcal{I}_\tau, \\ q_{ij} & \text{if } i \in \{\mathcal{I}'_\tau, \cup \mathcal{I}'_\theta\}, \\ 0 & \text{otherwise.} \end{cases} \quad (57)$$

Namely, we can write

$$\begin{aligned} g_u(\theta) &= \mathbb{E}[Y'] - \mathbb{E}[Y] = \sum_{i \in \mathcal{I}} w_i (q'_i - q_i) \\ &= \sum_{i \in \mathcal{I}_\tau} w_i q_i + \sum_{i \in \{\mathcal{I}'_\tau, \cup \mathcal{I}'_\theta\}} \sum_{j \in \mathcal{J}_i} w_{ij} q_{ij} - \sum_{i \in \mathcal{I}} w_i q_i \\ &= \sum_{i \in \{\mathcal{I}'_\tau, \cup \mathcal{I}'_\theta\}} \sum_{j \in \mathcal{J}_i} w_{ij} q_{ij} - \sum_{i \in \{\mathcal{I}_\tau, \cup \mathcal{I}_\theta\}} w_i q_i \\ &\leq \sum_{i \in \{\mathcal{I}'_\tau, \cup \mathcal{I}'_\theta\}} \sum_{j \in \mathcal{J}_i} w_{ij} q_{ij} \\ &\leq \sum_{i \in \{\mathcal{I}'_\tau, \cup \mathcal{I}'_\theta\}} \sum_{j \in \mathcal{J}_i} w_{ij} f(\theta) \leq f(\theta). \end{aligned} \quad (58)$$

The expression in (49) is Bernstein's inequality applied to the random variable  $Z_u = \sum_{i \in \mathcal{U}''} Z_i$  relying upon Lemma 11. Here, the concentration variables are  $Z_i = w_i Y_i - \mathbb{E}[w_i Y_i]$  with  $i \in \mathcal{U}''$  and we set  $\alpha_u = \tau'$ .

Let  $V_{\mathcal{U}''}$  be variance proxy term  $V$  in Bernstein's inequality as defined in (9) attached to  $\mathcal{U}''$ . The functions  $f, g : (0, 1) \times \mathbb{N} \rightarrow (0, 1)$  with  $f(x, n) = x(1-x)^n(1-(1-x)^n)$

and  $g(x, n) = x^2(1-x)^n(1-(1-x)^n)$  are non-increasing with respect to  $x$  on  $(\frac{1}{n+1}, 1)$  and  $(\frac{2}{n+2}, 1)$  respectively. We obtain for  $1 < \theta < n$ , an upperbound on  $V_{\mathcal{U}''}$  as follows:

$$\begin{aligned} V_{\mathcal{U}''} &= \sum_{i: w_i \in \mathcal{U}''} w_i^2 (1 - w_i)^n \left(1 - (1 - w_i)^n\right) \\ &\leq \tau \cdot \sum_{i: w_i \in \mathcal{U}''} w_i (1 - w_i)^n \left(1 - (1 - w_i)^n\right) \\ &= \tau \cdot \sigma_{\mathcal{U}''}^2 \\ &\leq \tau \cdot \sum_{i: \tau \leq w_i < \tau'; \sum_i w_i = 1} w_i (1 - w_i)^n \\ &\leq \underbrace{|\mathcal{I}(\theta, n)|}_{\leq \frac{n}{\theta}} \cdot \left(\frac{\theta}{n}\right)^2 \cdot \left(1 - \frac{\theta}{n}\right)^n \\ &\leq \frac{\theta}{n} \cdot e^{-\theta} < \frac{\theta}{n} \cdot \epsilon. \end{aligned} \quad (59)$$

In order to see why (52) holds, consider  $c(\gamma, \epsilon) = \frac{\epsilon(\gamma-1)^2}{\gamma^2 \ln(\frac{\gamma}{\epsilon})}$  and let us examine the derivatives as follows

$$\frac{\partial c(\gamma, \epsilon)}{\partial \gamma} = -\frac{\epsilon^2(\gamma-1)(\gamma-1-2\ln(\frac{\gamma}{\epsilon}))}{\gamma^3 \ln^2(\frac{\gamma}{\epsilon})}, \quad (60)$$

$$\begin{aligned} \frac{\partial^2 c(\gamma, \epsilon)}{\partial \gamma^2} &= \frac{\epsilon^2}{\gamma^4 \ln^3(\frac{\gamma}{\epsilon})} \left[ (6-4\gamma) \ln^2\left(\frac{\gamma}{\epsilon}\right) + \right. \\ &\quad \left. (\gamma^2 - 6\gamma + 5) \ln\left(\frac{\gamma}{\epsilon}\right) + 2(\gamma-1)^2 \right]. \end{aligned} \quad (61)$$

Solving for the first derivative using (60), we obtain

$$\gamma_\epsilon = -2W_{-1}\left(-\frac{\epsilon}{2\sqrt{e}}\right). \quad (62)$$

Inspecting the second derivative given by (61), we can see that the function  $c(\gamma, \epsilon)$  is concave with respect to  $\gamma$  for any  $\gamma > 2$ . Recall, moreover, that there are interrelated restrictions on  $\gamma, \epsilon$  and  $n$  in derivation of (51) and (52) which are collectively expressed as

$$\max\{e \cdot \epsilon, 1, 2, \gamma(1)\} < \gamma < e^n, \quad n \geq \lceil \gamma_\epsilon \rceil - 1. \quad (63)$$

## 5.2 Proof of Theorem 2: Lower Deviation Bound

The proof for lower deviation bound proceeds in the same spirit as section 5.1. The idea is again to reduce the problem to one in which all bins that are larger than the threshold  $\tau$  are eliminated.

We split large bins and then absorb small bins to enable us shrink the variance while controlling the magnitude of terms (and consequently the key quantities  $\alpha$  and  $V$ ) before applying Bernstein's inequality.

By choosing  $\theta$  such that  $f(\theta) = e^{-\theta}$  so that  $\theta = \ln(\frac{\gamma}{\epsilon})$ , for any  $0 < \epsilon < 1$  with  $e\epsilon < \gamma < e^n \epsilon$  being generic domain

for  $\gamma$  we obtain a lower deviation bound as follows

$$\mathbb{P}(Y - \mathbb{E}[Y] \leq -\epsilon) \leq \quad (64)$$

$$\mathbb{P}(Y' - \mathbb{E}[Y] \leq -\epsilon) = \quad (65)$$

$$\mathbb{P}(Y' - \mathbb{E}[Y'] + (\mathbb{E}[Y'] - \mathbb{E}[Y]) \leq -\epsilon) \leq \quad (66)$$

$$\mathbb{P}(Y' - \mathbb{E}[Y'] - f(\theta) \leq -\epsilon) = \quad (67)$$

$$\mathbb{P}\left(Y' - \mathbb{E}[Y'] \leq -\left(\frac{\gamma-1}{\gamma}\right)\epsilon\right) \leq \quad (68)$$

$$\leq \exp\left(-\frac{\left(\frac{\gamma-1}{\gamma}\right)^2 \epsilon^2}{2(V_{\mathcal{L}''} + \frac{\alpha_l}{3} \cdot \left(\frac{\gamma-1}{\gamma}\right) \cdot \epsilon)}\right) \leq \quad (69)$$

$$\leq \exp\left(-\frac{\left(\frac{\gamma-1}{\gamma}\right)^2 \epsilon^2}{2\left(\frac{\theta}{n} \cdot \epsilon + \frac{2\theta}{3n} \cdot \left(\frac{\gamma-1}{\gamma}\right) \cdot \epsilon\right)}\right) \leq \quad (70)$$

$$\inf_{\gamma} \left\{ \exp\left(-\frac{3n\epsilon(\gamma-1)^2}{10\gamma^2 \ln\left(\frac{\gamma}{\epsilon}\right)}\right) \right\} = \quad (71)$$

$$e^{-c(\epsilon) \cdot n\epsilon}, \quad (72)$$

where  $c(\epsilon)$  and  $\tau^*$  are as before and domain restrictions are determined similar to (63).

The variables  $Y'$  and  $Y''$ , and the sets  $\mathcal{L}'$  and  $\mathcal{L}''$  are defined in the same fashion as Section 5.1.

The first inequality is proved in the same way as (45). Now, we set  $\mathbb{E}[Y'_i] = q'_i$  such that

$$q'_i = \begin{cases} q_i & \text{if } w_i < \tau', \\ 0 & \text{otherwise.} \end{cases} \quad (73)$$

Inequality (67) follows because the compensation gap will remain small since we have

$$\begin{aligned} g_l(\theta) &= \mathbb{E}[Y'] - \mathbb{E}[Y] = \sum_{i \in \mathcal{I}} w_i (q'_i - q_i) \\ &= \sum_{i: w_i < \tau'} w_i q_i - \sum_{i \in \mathcal{I}} w_i q_i = - \sum_{i: w_i \geq \tau'} w_i q_i \\ &\geq - \sum_{i: w_i \geq \tau'} w_i f(\theta) \geq -f(\theta). \end{aligned} \quad (74)$$

The expression given by (69) is Bernstein's inequality applied to random variable  $Z_l = \sum_{i \in \mathcal{L}''} Z_i$  where we have defined  $Z_i = w_i(\mu - w_i Y_i) - \mathbb{E}[w_i(\mu - w_i Y_i)]$  with  $\mu$  being the upper bound on the value of the  $w_i Y_i$  terms.

Further, we choose  $\alpha_l = \tau'$ . Observe that  $Z_l = -Z_u$  and  $\mu = \alpha_l$ . Finally, an upperbound on  $V_{\mathcal{L}''}$  can be determined with arguments identical to that of  $V_{\mathcal{L}'}$ .

The rest of the proof proceeds in an analogous manner to the proof of upper deviation bound.

## 6 CONCLUSIONS

We proposed a new technique for establishing concentration inequalities and applied it to the missing mass using

Bernstein's inequality. Along the way, we introduced a collection of concepts and tools in the intersection of probability theory and information theory that have the potential to be advantageous in more general settings.

Recall that Bernstein's inequality hinges on establishing an upperbound on  $Z(X, \lambda)$  given by (29) in a particular way. Clearly, this choice is not unique and one can choose any other upperbound (e.g. c.f. Lugosi (2003)) and apply the same technique to derive potentially tight bounds achievable within the framework of exponential moment method.

Our bounds sharpen the leading results for missing mass in the case of small deviations. These inequalities hold subject to the mild condition that the sample size is large enough, namely  $n \geq \lceil \gamma \epsilon \rceil - 1$ .

We select the best known bounds in Berend and Kontorovich (2013) for the comparison. Our lower deviation and upper deviation bounds improve state-of-the-art for any  $0 < \epsilon < 0.021$  and any  $0 < \epsilon < 0.045$  respectively.

Plugging in the definitions, we can see that the compensation gap can be expressed as a function of  $\epsilon$  and show that the following holds

$$|g(\epsilon)| \leq \sqrt{\epsilon} \cdot \exp\left(W_{-1}\left(\frac{-\epsilon}{2\sqrt{\epsilon}}\right)\right), \quad (75)$$

where we have dropped the subscript of gap  $g$ . Note that the gap is negligible for small  $\epsilon$  compared to large values of  $\epsilon$  for both (52) and (72). This observation supports the fact that we obtained sharper bounds for small deviations.

Mathematical analysis of missing mass via concentration inequalities has various important applications including density estimation, generalization bounds and handling missing data just to name a few. Needless to say that any refinement in bounds or tools developed for the former may directly contribute to advancement in those applications.

## Acknowledgements

The authors are grateful to National ICT Australia (NICTA) for generous funding, as part of collaborative machine learning research projects. We would like to thank Peter Grünwald, Aryeh Kontorovich, Thijs van Ommen and Mark Schmidt for feedback and helpful discussions, and anonymous reviewers for their constructive comments.

## References

- Shun-ichi Amari. Divergence function, information monotonicity and information geometry. *Workshop on Information Theoretic Methods in Science and Engineering (WITMSE)*, 2009.
- Daniel Berend and Aryeh Kontorovich. The missing mass problem. *Statistics & Probability Letters*, 2012.

- Daniel Berend and Aryeh Kontorovich. On the concentration of the missing mass. *Electronic Communications in Probability*, 18:no. 3, 1–7, 2013.
- S. N. Bernstein. On a modification of Chebyshev’s inequality and of the error formula of Laplace. *Annals Science Institute SAV. Ukraine*, 1924.
- S. Boucheron, G. Lugosi, and P. Massart. *Concentration Inequalities: A Nonasymptotic Theory of Independence*. Oxford University Press, 2013.
- H. Chernoff. A measure of the asymptotic efficiency of tests of a hypothesis based on the sum of observations. *Annals of Mathematical Statistics*, 23:493–507, 1952.
- R. M. Corless, G. H. Gonnet, D. E. G. Hare, D. J. Jeffrey, and D. E. Knuth. On the Lambert W function. In *Advances in Computational Mathematics*, 1996.
- Imre Csiszár. Information measures: a critical survey. *7th Prague Conference on Information Theory*, pages 73–86, 1977.
- Imre Csiszár. Axiomatic characterizations of information measures. *Entropy*, 10:261–273, 2008.
- Devdatt Dubhashi and Desh Ranjan. Balls and bins: A study in negative dependence. *Random Structures and Algorithms*, 13:99–124, 1998.
- Kumar Joag-Dev and Frank Proschan. Negative association of random variables with applications. *Annals of Statistics*, 11:286–295, 1983.
- Michael Kearns and Lawrence Saul. Large deviation methods for approximate probabilistic inference. In *Proceedings of Uncertainty in Artificial Intelligence (UAI)*, 1998.
- Gábor Lugosi. Concentration of measure inequalities, 2003. URL <http://www.econ.upf.es/~lugosi/anu.ps>.
- David McAllester and Luis Ortiz. Concentration inequalities for the missing mass and for histogram rule error. *Journal of Machine Learning Research (JMLR)*, 4, 2003.
- David McAllester and Robert E. Schapire. On the convergence rate of Good-Turing estimators. *Conference on Learning Theory (COLT)*, 2000.
- Robin Pemantle. Towards a theory of negative dependence. *Journal of Mathematical Physics*, 41:1371–1390, 1999.

---

# Minimizing Expected Losses in Perturbation Models with Multidimensional Parametric Min-cuts

---

**Adrian Kim, Kyomin Jung**  
Seoul National University  
Seoul, South Korea  
{adkim955, kjung}@snu.ac.kr

**Yongsub Lim**  
KAIST  
Daejeon, South Korea  
ddiyong@kaist.ac.kr

**Daniel Tarlow, Pushmeet Kohli**  
Microsoft Research  
Cambridge, UK  
{dtarlow, pkohli}@microsoft.com

## Abstract

We consider the problem of learning perturbation-based probabilistic models by computing and differentiating expected losses. This is a challenging computational problem that has traditionally been tackled using Monte Carlo-based methods. In this work, we show how a generalization of parametric min-cuts can be used to address the same problem, achieving higher accuracy and faster performance than a sampling-based baseline. Utilizing our proposed *Skeleton Method*, we show that we can learn the perturbation model so as to directly minimize expected losses. Experimental results show that this approach offers promise as a new way of training structured prediction models under complex loss functions.

## 1 INTRODUCTION

Many problems in machine learning can be formulated as structured-output prediction, such as pixel labelling problems in computer vision and protein side-chain prediction in bio-informatics. A key challenge in the solution of these problems is to build structured prediction models that capture key correlations within the outputs and to learn these models from data. There are a range of approaches to this problem, including training a deterministic predictor to minimize (regularized) empirical risk (e.g., structural SVMs (Taskar et al., 2003; Tsochantaridis et al., 2005)), PAC Bayesian-based approaches where the goal is to train a randomized predictor to minimize a regularized empirical risk (Keshet et al., 2011), and probabilistic modelling paired with Bayesian decision theory (Schmidt et al., 2010).

Perturbation models (Papandreou & Yuille, 2011; Tarlow et al., 2012; Hazan & Jaakkola, 2012) are an approach that have been a focus of interest in recent years, and are closely

related to both PAC-Bayesian approaches and probabilistic modelling. The idea is to build a probabilistic model over structured outputs by drawing a random energy function and then returning the argmin of the random energy function as a sample from the model. These models can then be trained under maximum likelihood-like objectives (Papandreou & Yuille, 2011; Tarlow et al., 2012; Hazan & Jaakkola, 2012) or to minimize expected loss (Keshet et al., 2011; Hazan et al., 2013). Typically the distribution over energy functions is restricted so that the optimization step is tractable (e.g., it is a min-cut problem). When this is the case, perturbation models have the desirable property that exact samples can be drawn efficiently with a single call to an efficient optimization procedure.

Our aim in this work is to revisit the problem of training perturbation models to minimize expected losses. Previous works (Keshet et al., 2011; Hazan et al., 2013) have used Monte Carlo-based methods to estimate the needed gradients. A concern with these approaches is that the gradient estimates can have high variance, as is the case with the well-known REINFORCE algorithm (Williams, 1992). Instead, our approach here is to explore combinatorial methods that take advantage of the structure of the optimization problem in order to more efficiently make use of optimizer runs. As a first foray into this approach, we restrict attention to the case where the perturbation model takes the form of a uniform distribution over model parameters followed by a call to a min-cut/maxflow routine.

Our method is based on a generalization of the parametric min-cut algorithm (Gallo et al., 1989) which in the 1-dimensional case is able to efficiently compute all parameter values (breakpoints) where the minimum energy (MAP) solution changes. To demonstrate the efficacy of our method, we compare estimated expected losses and their gradients computed by our method with those obtained from a sampling-based scheme. Experimental results show that we get more accurate solutions with fewer calls to the optimization procedure and less overall wall time.

As a full application, we also show that our method is use-

ful towards training structured prediction models to minimize expected losses. The method is indifferent to the loss function used, so there is potential to use the same method for loss functions that are typically difficult to work with. Experimentally, we compare our method to learning using Perturb-and-MAP (a.k.a. P&M) (Papandreou & Yuille, 2011) to learn a probabilistic model, then making loss-aware predictions using Bayesian Decision theory. We also show that the Skeleton method can be used in place of sampling within the training procedure from (Papandreou & Yuille, 2011) to give gradients with lower variance.

## 2 BACKGROUND: PERTURBATIONS, EXPECTED LOSSES

We will focus on the case where perturbation models are used to define a conditional probability model  $P(y|x; \theta)$ , where  $x$  is an input (e.g., an image),  $y \in \{0, 1\}^n$  is a structured output (e.g., a foreground-background image segmentation), and  $\theta \in \mathbb{R}^m$  is a real-valued vector of parameter values. We additionally assume access to a feature vector  $\phi(x, y) \in \mathbb{R}^m$  which contains unary and pairwise potentials. Perturbation models begin by defining an energy function  $E(y|x; \theta) = \langle \theta, \phi(x, y) \rangle$ . The second component to a perturbation model is the noise distribution  $P(\gamma)$  which is a distribution over noise vectors  $\gamma \in \mathbb{R}^m$ . The probabilistic model  $P(y|x; \theta)$  can then be defined as follows:

$$\gamma \sim P(\gamma) \quad (1)$$

$$y = \operatorname{argmin}_{y'} E(y'|x; \theta + \gamma). \quad (2)$$

It will be useful to define *minimizer*  $f(\theta) = \operatorname{argmin}_y E(y|x; \theta)$ , *dual function*  $g(\theta) = \min_y E(y|x; \theta)$ , and *inverse set*  $f^{-1}(y) = \{\theta : f(\theta) = y\}$ . Under this definition, the probability of a configuration  $y$  can be expressed as  $P(y|x; \theta) = \int \mathbf{1}_{\{\theta + \gamma \in f^{-1}(y)\}} P(\gamma) d\gamma$ . We are interested in expected losses under perturbation models. The expected loss (or *risk*) is a function of a given  $y^*$  (in our case, the ground truth configuration) and parameters  $\theta$ . It is defined as

$$\begin{aligned} R(y^*, \theta) &= \sum_{y \in \{0, 1\}^n} P(y|x; \theta) L(y^*, y) \quad (3) \\ &= \sum_{y \in \{0, 1\}^n} \int \mathbf{1}_{\{\theta + \gamma \in f^{-1}(y)\}} P(\gamma) L(y^*, y) d\gamma, \quad (4) \end{aligned}$$

where  $L(y^*, y)$  assigns a loss value for predicting  $y$  when the ground truth is  $y^*$ . The ultimate goal we are working towards is to learn parameters  $\theta$  so as to minimize  $R(y^*, \theta)$ . First, we focus on the prerequisite tasks of computing and differentiating  $R(y^*, \theta)$ . We will use deterministic update rules to calculate gradients with the Skeleton method to learn the model.

## 3 ALGORITHM: SKELETON METHOD

We begin by making some assumptions. First, let  $P(\gamma)$  be a uniform distribution such that  $\theta + \gamma$  is distributed uniformly over a  $m$ -dimensional hyperrectangular region  $S_\theta = \prod_{i=1}^m [\theta_i, \theta_i + w_i]$ , where  $\gamma_i \in [0, w_i]$  and  $w_i \in \mathbb{R}^{>0}$ . Also assume that the minimizer  $f(\theta)$  is unique for all  $\theta$  except for a set with measure zero, so  $f(\theta)$  can be treated as having a unique value. Finally, assume that for all  $\theta \in S$ ,  $E(y|x; \theta)$  is submodular and can be optimized efficiently.

In the following, it will be convenient for us to redefine the inverse set  $f^{-1}(y)$  so that only regions in  $S$  are included. That is,  $f^{-1}(y) = \{\theta : f(\theta) = y \wedge \theta \in S\}$ . Then from above, we have that  $R(y^*, \theta) = \sum_{y \in \{0, 1\}^n} \int \mathbf{1}_{\{\theta + \gamma \in f^{-1}(y)\}} P(\gamma) L(y^*, y) d\gamma$ . Noting that  $L(y^*, y)$  is not a function of  $\gamma$  and that  $\int \mathbf{1}_{\{\theta + \gamma \in f^{-1}(y)\}} P(\gamma) d\gamma = \text{Volume}(f^{-1}(y)) / \text{Volume}(S)$ , we can rewrite  $R(\cdot)$  as  $\sum_{y \in Y_S} L(y^*, y) \text{Volume}(f^{-1}(y)) / \text{Volume}(S)$ , where  $Y_S = \{y : \exists \theta, \theta \in S \wedge f(\theta) = y\}$  is the set of configurations that are minimizers for some  $\theta \in S$ .

In this paper, we introduce a novel method to find the minimizers  $y \in Y_S$  and their inverse sets by iteratively updating a graph structure that we call a *skeleton*. Note that for a fixed  $y$ ,  $E(y|x; \theta)$  is a linear function of  $\theta$ , which implies that the dual function  $g(\theta) = \min_y E(y|x; \theta)$  is a piecewise concave function, where pieces are hyperplanes corresponding to minimizers  $y$ . Let  $h_y$  be the corresponding hyperplane for some fixed minimizer  $y$ . Intuitively, the skeleton  $G_Y = (V_Y, E_Y)$  is a graphical representation of the dual  $g(\theta)$  over  $S$ . The skeleton will be constructed on the given parameter space  $S$  by finding new minimizers, or hyperplanes, at each iteration until there are no more minimizers. At each iteration, the growing skeleton represents an upper bound on the dual  $g$ , which we call the *subset dual*.

**Definition 1 (Subset dual  $g_Y$ )** For some given minimizer set  $Y \subseteq Y_S$ , let  $g_Y(\theta) = \min_{y \in Y} E(y|x; \theta)$  be the subset dual, which is a piecewise concave function.

For some given subset dual  $g_Y(\cdot)$ , each hyperplane  $h_y$  has a corresponding graph which we refer as a *facet*  $G_y$ . A facet  $G_y = (V_y, E_y)$  is defined as the smallest convex hull made by the intersections of  $h_y$  and other hyperplanes, where  $V_y, E_y$  are boundary vertices and edges of the convex hull. Let  $\theta_v$  be the parameter value and  $z_v = g_Y(\theta_v)$  be the subset dual value corresponding to the vertex  $v$ . Note that a facet can be cut because of the boundaries the given parameter space makes. A skeleton is defined using the union of these facets as follows.

**Definition 2 (Skeleton of  $g_Y$  over  $S$ )** For some given subset dual  $g_Y$ , the skeleton of  $g_Y$  on  $S$  can be represented by the following structure  $G_Y = (V_Y, E_Y)$ . Let  $(u, v)$  be

an edge between  $u$  and  $v$ , where  $u, v \in V_Y$ .

- $V_Y = \bigcup_{y \in Y} \{v : \text{Boundary vertices of } G_y, \text{ where } \theta_v \in S, z_v = g_Y(\theta_v)\}$
- $E_Y = \bigcup_{y \in Y} \{(u, v) : \text{Boundary edges of } G_y, \text{ where } u, v \in V_Y\} \cup \{(u, v) : u \in V_Y, \theta_u \in \Pi_{i=0}^m \{w_i^-, w_i^+\}, v = (\theta_u, -\infty)\}$

For example, Figure 1 is a skeleton over some parameter space  $S \in \mathbb{R}^2$  given a subset dual  $g_Y$ , where  $Y = \{y_1, \dots, y_5\}$ . There are five facets on the skeleton, where four are cut by the boundaries of  $S$ .

From the given definitions, it is clear that an inverse set  $f_Y^{-1}(y) = \{\theta : y = \operatorname{argmin}_{y' \in Y} f_\theta(y') \wedge \theta \in S\}$  of  $y$  defined on a subset dual  $g_Y$  directly corresponds to the projection of the facet  $G_y$ . Thus, in order to calculate the volume of  $\theta_Y(y)$ , we can use the projected vertices of  $G_y$  on  $S$ . One of the main points of our method is that we are able to track every facet with every iteration, so that we can calculate the approximate expected loss every time we update the skeleton.

We now describe our *Skeleton method* and how it works. Figure 1 describes a visual example on how a skeleton is constructed and updated by a single iteration of our algorithm. Algorithm 1 is the pseudo code of the algorithm.

### 3.1 INITIALIZATION

The initial skeleton  $G_Y = (V_Y, E_Y)$  is given by the following.

- $Y = \phi$
- $V_Y = \{(\theta_{v_n}, z_{v_n}) : \theta_{v_n} = \Pi_{i=0}^m \{w_i^-, w_i^+\}, z_{v_n} = \infty\}$
- $E_Y = \{(u, v) : u \in V_Y, v = (\theta_u, -\infty)\}$

### 3.2 FINDING A NEW FACET

In order to find a new facet, the algorithm first picks some vertex  $u = (\theta_u, z_u) \in V_Y$ . Using graph cut, a new solution  $y_u = f(\theta_u)$  can be found. The first step is to determine whether the new solution improves the current dual in any region. This can be checked by  $h_{y_u}(\theta_u) < z_u$ . If this is the case, we say that a *cut* is made, and  $y_u$  is added to  $Y$ .

Next, we must find new intersection points where  $h_{y_u}$  intersects other hyperplanes defining the subset dual. The key property of the new intersection points is that they will either appear at existing vertices  $v \in V_Y$ , or they appear on an edge  $(p_h, p_t) \in E_Y$  that ‘‘crosses’’ the new hyperplane; that is  $h_{y_{p_h}}(\theta_{p_h}) < z_{p_h}$  and  $h_{y_{p_t}}(\theta_{p_t}) > z_{p_t}$ . The set of vertices where  $h_{y_v}(\theta_v) < z_v$ , form a connected component

---

### Algorithm 1 Skeleton Method

---

**Input:** Oracle  $f$ , Loss function  $L(y^*, y)$   
 $(Y, G_Y) \leftarrow \text{InitSkeleton}()$   
**for all**  $u = (\theta_u, z_u) \in V_i$  **do**  
     $y_u = f(\theta_u)$   
    **if**  $h_{y_u}(\theta_u) < z_u$  **then**  
        Add  $y_u$  to  $Y$   
         $(I, H) \leftarrow \text{FindIntersection}(G_Y, h_{y_u})$   
        Add  $f_{y_u} = (y_u, I)$  to  $F_Y$   
         $V_Y = (V_Y \cup I) - H$   
        **for all** Intersection vertices  $p \in I$  **do**  
            **if**  $p$  is a new vertex **then**  
                Add  $p$  to all  $G_y \in \{\text{Facets sharing } (p_t, p_h), \text{ where } p_h \text{ is above and } p_t \text{ is below } h_{y_u}\}$   
                Append new edge  $(p_t, p)$  to  $E_Y$   
            **end if**  
        **end for**  
        Remove vertices  $r \in H$  above  $h_{y_u}$  from all facets  
        Remove  $E_- = \{(u, v) : u \text{ or } v \in H\}$  from  $E_Y$   
        Append  $E_+ = \{\text{Boundary edges of } G_{y_u}\}$  to  $E_Y$   
         $R = \sum_{y \in Y} \text{Volume}_{f^{-1}(y)} L(y^*, y)$   
    **end if**  
**end for**

---

$H \subset G_Y$ , and the crossing edges are the boundary edges of this connected component. Thus, the intersection points can be found by exploring a search tree outwards from  $u$ . When a vertex  $v$  is encountered such that  $h_{y_v}(\theta_v) < z_v$ , the intersection point between  $v$  and its parent is computed by finding some point  $p$  where  $h_{y_p}(\theta_p) = z_p$ , and the search tree is not searched further down that path. Upon termination, vertices of the connected component  $H$  are removed from  $V_Y$ , and the new intersection points, notated as  $I$ , are added. A step of this procedure is illustrated in Figure 1(b), where there is a cut after selecting vertex  $u_i$ , colored in red.

### 3.3 UPDATING THE SKELETON $G_Y$

When a cut is done in the skeleton, it should be updated with the new upper bound made by  $h_{y_u}$ . The nontrivial case is when some intersection point  $p \in I$  is a new point made on some edge  $(p_h, p_t) \in E_Y$ , which is  $(u_1, v_1)$  for  $p_1$  in Figure 1b. The new vertex  $p$  is added to all facets which share the edge  $(p_h, p_t)$ . Also, a new edge  $(p_t, p)$  should be added to the skeleton. Boundary edges made from the convex hull of the new polytope  $G_{y_u}$  are also added to  $E_Y$ . Finally, the skeleton update is done when all vertices  $r \in H$  are deleted from every facet and all edges including  $r$  are removed from  $E_Y$ .

### 3.4 CALCULATING EXPECTED LOSS $R$

At this point, the skeleton is fully updated. To compute expected loss  $R(y^*, \theta)$ , we use an off-the-shelf subroutine

for computing the volume of each inverse set  $f_Y^{-1}(y)$  for  $y \in Y$ . The volumes are multiplied by the loss value for each  $y$ , and the products are summed to get the full expected loss. For normalization, the value is divided by the volume of  $S$ .

### 3.5 EXAMPLE : TWO PARAMETERS

Figure 1 describes a single iteration of the Skeleton Method on a perturbation model having two parameters,  $\theta_1, \theta_2$ . Note that the leftside represents the subset dual  $g_Y$  and that the right image is the projection of facets on the given parameter space  $S$ . The iteration starts from a skeleton which has already done five iterations by the algorithm ( $Y = \{y_1 \dots y_5\}$ ). There are five facets on the parameter space so that the expected loss is  $R(y^*, \theta) = \sum_{n=1}^5 \text{Volume}(f_Y^{-1}(y_n))L(y^*, y_n) / \text{Volume}(S)$ . Suppose we take some unused vertex  $u_1$ . In this case, we can see that the hyperplane  $h_{y_6}$  makes a cut in the skeleton ( $Y' = Y \cup \{y_6\}$ ). By updating the skeleton, a new facet  $f_6$  is found. Since there is a unique loss value for each facet, we can calculate the expected loss as  $R(y^*, \theta) = \sum_{n=1}^6 \text{Volume}(f_{Y'}^{-1}(y_n))L(y^*, y_n) / \text{Volume}(S)$ .

## 4 LEARNING

### 4.1 COMPUTING GRADIENTS: SLICING

Our main focus is not only computing expected losses; the ultimate aim is to learn parameters that yield a perturbation model that achieves low expected loss. In order to update the perturbation model to minimize the expected loss, we calculate the gradients by applying a simple finite-differencing-based technique named **slicing**.

Before going through details, we add more assumptions from the previous section. To be more flexible, let the parameter space where  $\theta + \gamma$  is sampled from notated as  $S_\theta = \theta + S = \theta + \Pi_i^m \gamma$ , where  $\gamma_i \in [0, w_i]$ . The expected loss of the region which  $S$  creates on parameter  $\theta$  will be notated as  $R_S(y^*, \theta)$ . The approximation we use is

$$\frac{\partial R(y^*, \theta)}{\partial \theta_i} \approx \frac{R_S(y^*, \theta + \delta e_i) - R_S(y^*, \theta)}{\delta}, \quad (5)$$

where  $\delta$  is a small value and  $e_i \in \mathbb{R}^m$  is a unit vector with 1 in the  $i$ th coordinate and 0 elsewhere. Intuitively, this is identical to the difference between expected losses of regions shifted to the + direction of  $\theta_i$  by a small distance of  $\delta$ . Therefore, we can use a Monte Carlo-based method or the Skeleton method on these two regions and compute the differences to find the gradients.

When shifting a region by  $\delta$ , we see that the contribution to the gradient comes just from the  $\delta$ -width end-regions illustrated in Figure 3 (b). We call these end regions *slices*. Motivated by this, instead of computing expected losses of

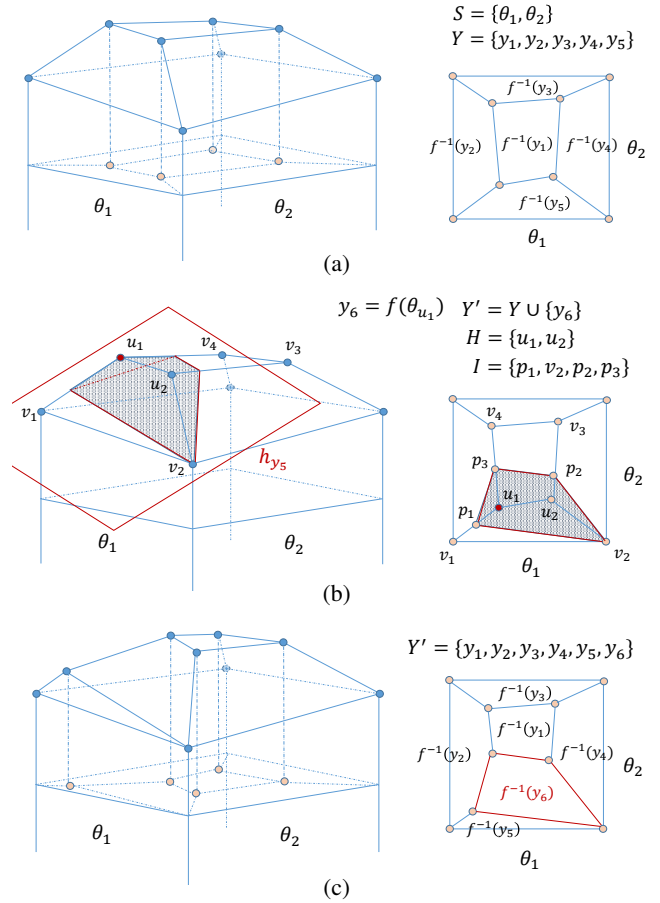


Figure 1: Visual Example of the Skeleton Method with Two Parameters.

the shifted and unshifted full regions, we compute expected losses only on the slices. Intuitively, we expect the slices to have fewer minimizers defining the Skeleton structure than the full regions that include them, and we expect that focusing only on the regions of difference will lead to faster and more accurate gradient estimates.

Let  $s^i = \Pi_j^m \gamma_j$  be the size of the thin slice where  $\gamma_j = [0, w_j]$  except the  $i$ th range  $\gamma_i \in [0, \delta]$ . From this setting,  $R_{s^i}(y^*, \theta)$  stands for the expected loss of the sliced region of size  $s^i$ . Using this we can apply gradient descent updates.

$$\frac{\partial R(y^*, \theta)}{\partial \theta_i} \approx R_{s^i}(y^*, \theta + w_i) - R_{s^i}(y^*, \theta) \quad (6)$$

$$\theta_i(t+1) = \theta_i(t) - \alpha_i \frac{\partial R(y^*, \theta)}{\partial \theta_i} \quad (7)$$

Each parameter  $\theta_i$  in iteration  $t+1$  is updated with the gradient value with a constant step size of  $\alpha_i$ , which is proportional to the feature size of  $\theta_i$ . One thing to be cautious about when selecting a learning rate is, that if the



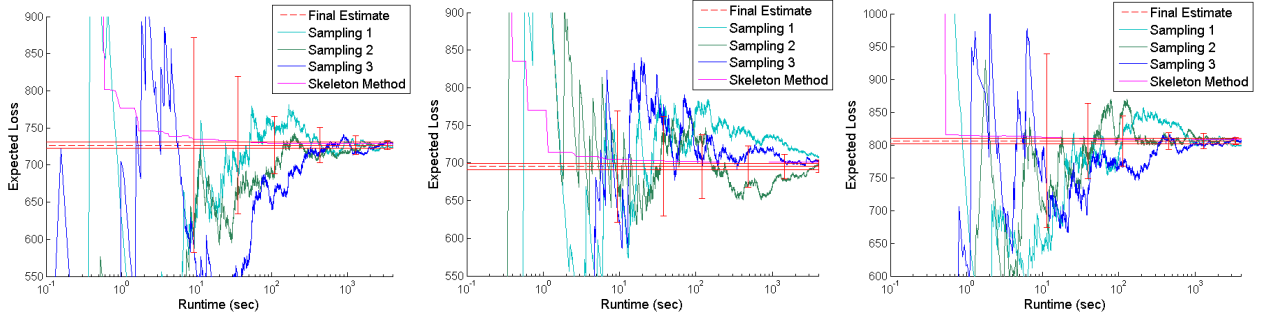


Figure 2: Comparisons between the Monte Carlo Estimator and the Skeleton Method. Using both methods we computed expected Hamming losses to identical settings and compared by runtime. Images of size [90x120] taken from (Rother et al., 2004) are used.

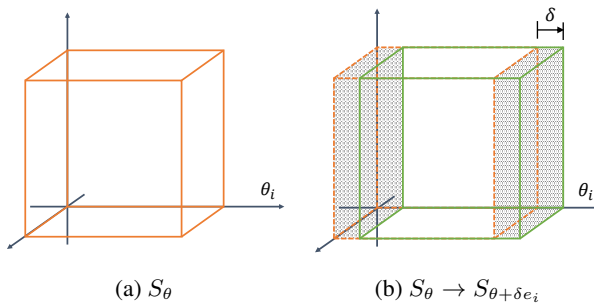


Figure 3: Visual Discription of Slices. The Slicing method computes the differences in gray regions in (b) to estimate the gradient with respect to parameter  $\theta_i$ .

learning rate is too large, then the parameters may make the model jump to an **unlearnable** state (plateau in the objective), which is a state where  $S_\theta$  holds only one inverse set.

## 4.2 TRAINING

In order to learn the parameters for our perturbation model, we exploit the Slicing method so that the model is trained directly from minimizing expected loss. One main advantage for our method is that we can use an arbitrary loss function very easily in this process. Suppose we have a training set with a size of  $N$  and have  $m$  parameters. For each iteration, we make  $2m$  slices from the model. Parameters are updated by using the mean value of gradients from all training images like the following equation.

$$\theta_i(t+1) = \theta_i(t) + \alpha_i \frac{1}{N} \sum_n \frac{\partial R(y_n^*, \theta)}{\partial \theta_i} \quad (8)$$

Note that it is not necessary to evaluate the expected loss objective at every step of the optimization.

## 4.3 EXPLOITING THE SKELETON METHOD

Previous approaches focus on how to use sampling methods to learn their models, which although many have well understood theoretical convergence properties as the sample size goes to infinity, suffer from problems with high variance in practice. In fact, the Skeleton method can be used in place of sampling more generally; for example, the P&M model in (Papandreou & Yuille, 2011) is trained using a moment-matching objective described in Eq. 9-11.

$$\theta_i(t+1) = \theta_i(t) - \alpha_i \Delta \theta_i \quad (9)$$

$$\Delta \theta_i = E_{S_\theta}[\phi_i(y)] - E[\phi_i(y^*)] \quad (10)$$

$$E_{S_\theta}[\phi_i(y)] = \frac{1}{M} \sum_j^M \phi_i(y_j) \quad (11)$$

To compute the expectations  $E_{S_\theta}[\phi_i(y)]$ , where  $\phi_i(y)$  is a feature function, the standard approach is to use sampling. However, we can replace the sampling-based approach with a skeleton-based approach. Specifically, we replace the term to be  $E_{S_\theta}[\phi_i(y)] = \sum_{y \in \{0,1\}^n} P(y|x, \theta) L_H(y^*, y_j)$  and then use the method described above to compute the quantities needed in Eq. 9-11. This gives an alternative method for optimizing the original P&M objective; we call this approach Skeleton Perturb-and-MAP (Skeleton P&M).

## 5 EXPERIMENTS AND DISCUSSION

### 5.1 DATA AND SETUP

In this section, we apply the Skeleton Method to a foreground-background image segmentation task, comparing against Monte Carlo baselines which estimate expected losses by drawing samples from the prior and reporting the average incurred loss. All images used in experiments are

originally from the Berkeley image segmentation set by (Rother et al., 2004). The energy function used is of the following form:

$$\begin{aligned}
 E(y | x; \theta) &= \langle \theta, \phi(x, y) \rangle \\
 &= E(x) + \theta_1 \sum_i^n x_i \\
 &\quad + \theta_2 \sum_{(y_i, y_j) \in E_v} (y_i \neq y_j) + \theta_3 \sum_{(y_i, y_j) \in E_h} (y_i \neq y_j)
 \end{aligned}
 \tag{12}$$

where  $E_v, E_h$  are each sets of neighboring vertical and horizontal pairs of pixels respectively.  $x_i$  is the  $i$ th pixel’s label of the noised input, which is made by switching values of ground truth labels with a uniform probability of 5%.

Expected losses were computed over a parameter space  $S_\theta = \theta + \gamma \subseteq \mathbb{R}^m$  defined from a uniform distribution  $\gamma \sim P(\gamma)$  where  $\gamma \in [0, 1]^3$ . Intuitively,  $S_\theta$  is a cube shaped region positioned by  $\theta$  on the parameter space where parameters are sampled from. Expected loss over region  $S_\theta$  will be notated as  $R_S(y^*, \theta)$ .

In default, the loss function for the following experiments will be defined as the Hamming distance,  $L_H(y^*, y) = \sum_{i=1}^n (y_i \neq y_i^*)$ . Note that this formulation supports arbitrary loss functions other than the Hamming distance.

## 5.2 CALCULATING EXPECTED LOSSES

To evaluate the methods, it would be ideal to have a ground truth value of expected losses for a given parameter setting. Unfortunately this is hard to calculate accurately, because the Skeleton method does not always run to termination within practical time, and there is necessarily some variance in the estimates returned by the sampling estimate. Thus, we report the estimates from each method along with 95% confidence intervals derived from the sampling method. For the sampling method, in each trial, parameters were independently sampled 100k times, and this was repeated 10 times.

Figure 2 shows plots of expected losses calculated by the two methods versus runtime. The average sampling estimate (across all trials) appear as red dashed lines in Figure 2(a) -2(c). Also shown are the cumulative averages for three representative trials of the sampling (green to blue curves), and the Skeleton method (magenta). The main take-away is that the expected loss values of both methods converge to similar values, but particularly with few samples, there is high variance in sampling. While the Monte Carlo estimator has significant variance even after 1000 seconds, the Skeleton Method has essentially converged to its final, accurate estimate after approximately 10 seconds. This suggests that we can even stop running the method in the middle of the algorithm to estimate the expected loss with high accuracy. The reason such behavior appears is

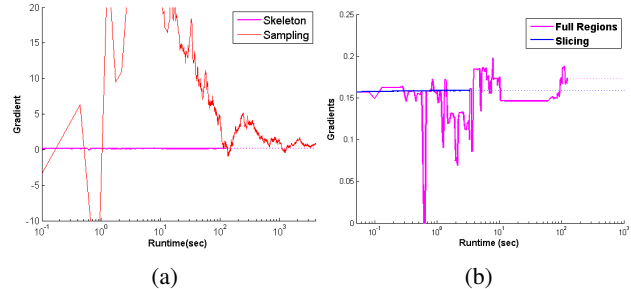


Figure 4: Gradients of Expected Hamming Loss for  $\theta_1$ . (a) Sampling and Skeleton method on two full regions (b) Skeleton method on full regions and Slicing method.

related to the high concentration of vertices in the later iterations of the algorithm. Many calculations made in later iterations induce inverse sets which have very small volumes, implying the low contribution to the expected loss.

## 5.3 CALCULATING GRADIENTS

We now turn attention to evaluating the Skeleton method and Monte Carlo method for computing gradients of expected losses. For the Skeleton method, we evaluate our recommended Slicing method, and also a variant that computes expected losses over full regions that are shifted by  $\delta$ , which would be the more standard finite-difference approach. We use the thickness  $\delta = 0.001$  and parameter  $\theta_1$ , which is for the unary term, for the experiments. A comparison of the Monte Carlo approach (red) and the full-region Skeleton method (magenta) appear in Figure 4 (a). The red curve shows the cumulative average Monte Carlo estimate averaged across 10 repetitions. Even with this averaging, we see a great deal of variance in the estimates. The Skeleton method, by contrast, quickly converges to a value near where the Monte Carlo estimator appears to be converging to.

We then zoom in (note the y-axis scales) and consider the recommended Slicing variant of the Skeleton method and compare it to the full-region version shown in Figure 4 (a). The result appears in Figure 4 (b). Here we see that the Slicing variant is faster and much more stable than the full-region variant. As mentioned above, we believe the reason for the disparity is that number of unique inverse sets in the Slicing variant is smaller, and there is no variance that arises from the two runs computing slightly different estimates of the expected loss in the middle region that is contained by both the original and shifted full region.

## 5.4 MODEL LEARNING

Learning was done on an image set including 30 images each having approximately 2500 pixels. The data set was randomly split into  $N = 24$  training images and  $N' = 6$  test

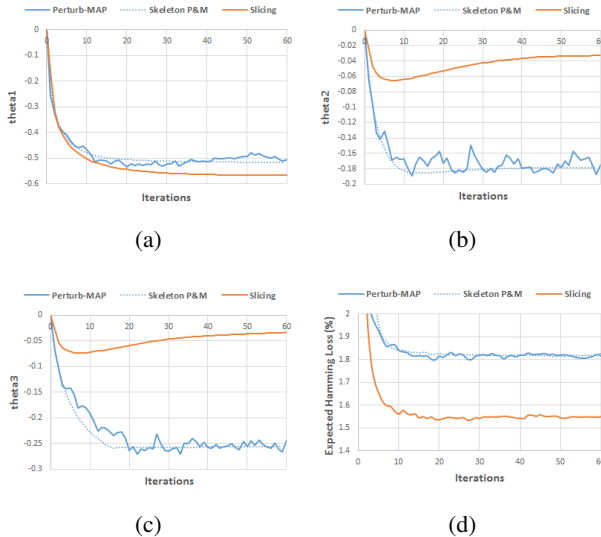


Figure 5: Parameter Learning with P&M and the Slicing Method. (a)-(c): Parameter updates (d): Expected Hamming Loss updates

images.

#### 5.4.1 Learning

We performed learning with the Slicing method, where gradients are computed with slices having a thickness of  $\delta = 0.001$ . The starting parameter is  $\theta = (0, 0, 0)$ , with a uniform perturbation  $\gamma \in \Pi_i^3[0, 1]$  defining a cube-shaped region on the parameter space. Gradients are computed for each parameter, which makes 6 slices to use. All slices can be computed independently, where in most cases 1-3 seconds are enough to get significant accuracy. By every iteration, the region will shift to a certain direction, and the process is repeated. The orange plots of Figure 5 show how the Slicing method learns the model for 60 iterations.

As a baseline for our method, we trained the P&M model with the same settings. Note that the Slicing method and P&M model have different behaviors, which are due to the difference in objectives; our Slicing method directly tries to minimize expected Hamming loss while the P&M model uses a moment-matching rule to estimate the posterior. The behavior of the learning P&M model is illustrated as the solid blue line of Figure 5, with the Skeleton P&M model being the dotted blue line. Take note that the Skeleton P&M strongly resembles the original P&M trace, but its trajectory is smoother, presumably due to lower variance in the gradient estimates.

At test time, instead of computing expected losses accurately, there may be a desire to sacrifice accuracy over runtime in estimating the value. One easy example is to use a finite number of samples such as 20 and compute the average of losses. Another approach is to sample a single output

Table 1: Expected Hamming Losses. Expected losses are computed with three ways 1) Average loss of 20 samples 2) Skeleton method 3) Single sampled loss from center. The performance for each model is described in each row, where values were computed separately on the training set and test set.

| METHOD            | SAMPLED            | EXPECTED     | CENTER       |
|-------------------|--------------------|--------------|--------------|
| P&M (Train)       | 1.694±.0011        | 1.812        | <b>.2369</b> |
| Skel. P&M (Train) | 1.764±.0017        | 1.816        | <b>.2369</b> |
| Slicing (Train)   | <b>1.480±.0012</b> | <b>1.535</b> | .2932        |
| P&M (Test)        | 2.186±.0011        | 2.257        | <b>1.172</b> |
| Skel. P&M (Test)  | 2.197±.0016        | 2.268        | 1.178        |
| Slicing (Test)    | <b>2.048±.0043</b> | <b>2.134</b> | 1.391        |

from a moderate position such as the center of the parameter space. Table 1 shows the expected losses computed from the mentioned methods. Each column represents the method we choose to compute expected loss. Each row represents the selected model trained for 60 iterations. Both from Figure 5(d) and Table 1, it is clear that our method is superior to the P&M model in optimizing the expected Hamming loss.

#### 5.4.2 Other Loss Functions

With our method, it is possible to minimize an arbitrary loss function’s expected value. In the following experiments we try to minimize the following loss function.

- **Boundary-only Pixel Loss**  $L_P$ : Hamming loss on only pixels which have at least one neighbor with a different label in the ground truth

$$L_P(y^*, y) = \sum_{y_i^* \neq y_j^*} (y_i \neq y_j)$$

The solid lines of Figure 6 shows the expected losses changing by the Slicing method in 60 iterations. The dashed lines are loss values from a baseline where we use the learned Skeleton P&M parameters to make loss-directed predictions using an approximation of Bayesian decision theory, similar to that used by (Premachandran et al., 2014). approximation Bayesian Decision Theory prediction framework. Specifically, we sample  $M = 100$  segmentations  $Y = \{y^{(1)}, \dots, y^{(M)}\}$  from the learned model, then we make predictions by restricting possible predictions to be one of the  $M$  sampled segmentations, and we approximate expected losses by taking averages over the  $M$  segmentations; specifically, we predict as  $\arg \min_{y' \in Y} \sum_{y \in Y} \Delta(y', y)$ . Figure 6 shows the expected boundary-only pixel loss being learned from the Slicing method as solid lines and the approximate Minimum Bayes Risk (MBR) prediction loss as dashed lines. This experiment shows that our method gives better results than the classical approach in minimizing expected losses.

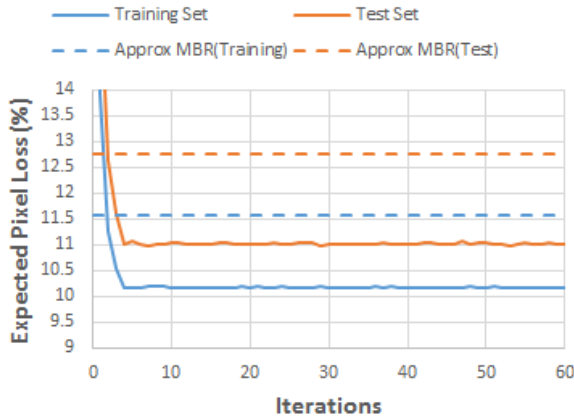


Figure 6: Learning Other Expected Losses. Orange and blue lines represent the values computed from the test set and training set respectively. Dashed lines are Approximate MBR prediction loss values, while the solid lines are learned from the Slicing method.

## 5.5 EXPECTED SEGMENTATIONS

To visualize how different parameters, or regions, effect the expected loss, we can use a probabilistic image constructed from every solution captured by our algorithm. This image or *expected segmentation* is made by weighting each configuration by the volume of its inverse set and summing up to a gray scale image. Note that this implies that the values of Table 1 are identical to the  $l_1$  distance between the ground truth image and the expected segmentation. Examples are shown in Figure 7. The example images were selected from the test set. From the figure, you can see that expected segmentations made from our perturbation model have higher quality, smoother segmentations than those from the P&M models.

## 6 CONCLUSION

Our results show that the Skeleton method is a promising alternative to Monte Carlo methods. The Skeleton method converges in a nice deterministic behavior, which shows higher accuracy than using samples. Another benefit of the Skeleton method is that it is applicable for any loss function. We have shown it applied to a boundary-only pixel loss; in future work it would be interesting to apply it to even more complicated loss functions. The Skeleton method also appears to be a general drop-in replacement for sampling-based computation of expectations in perturbation models. We showed this by adapting the method to the moment-matching objective that the original P&M paper proposed, showing that the Skeleton method leads to similar-but-smoother learning trajectories.

The idea of iteratively building piecewise linear approxi-

mations arises in many cases, such as when computing the value function in POMDPs (Porta et al., 2006; Isom et al., 2008; Brechtel et al., 2013). While the high level ideas are similar to these and other methods, the details are quite different; for example, in the above works, no volume computations are required, whereas they are core to our method.

The primary challenge going forward is to broaden the applicability of the method, extending to higher dimensions and enlarging the space of supported perturbation distributions. It is likely in these cases that exactness of the method will need to be abandoned due to the fact that the number of solutions will likely grow, and the computations of necessary volumes will become computationally hard. Despite this, we believe the algorithm presented here will be useful going forward. There are two possibilities we are interested in exploring: first, using a hybrid of the Skeleton and sampling methods where some dimensions are sampled and some are integrated analytically using the Skeleton method (producing a Rao-Blackwellized sampler); second, we believe there to be opportunities for computing and differentiating upper bounds based on the Skeleton structure, which could lead to interesting new learning methods.

## Acknowledgements

K.Jung is with ASRI, Seoul National University, and he is funded by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (2012R1A1A1014965).

## References

- Brechtel, Sebastian, Gindele, Tobias, et al. Solving continuous pomdps: Value iteration with incremental learning of an efficient space representation. In *Proceedings of the 30th International conference on machine learning*, pp. 370–378, 2013.
- Gallo, Giorgio, Grigoriadis, Michael D, and Tarjan, Robert E. A fast parametric maximum flow algorithm and applications. *SIAM Journal on Computing*, 18(1): 30–55, 1989.
- Hazan, Tamir and Jaakkola, Tommi S. On the Partition Function and Random Maximum A-Posteriori Perturbations. In *ICML*, pp. 991–998, 2012.
- Hazan, Tamir, Maji, Subhransu, Keshet, Joseph, and Jaakkola, Tommi. Learning efficient random maximum a-posteriori predictors with non-decomposable loss functions. In *Advances in Neural Information Processing Systems*, pp. 1887–1895, 2013.
- Isom, Joshua D, Meyn, Sean P, and Braatz, Richard D. Piecewise linear dynamic programming for constrained pomdps. In *AAAI*, pp. 291–296, 2008.

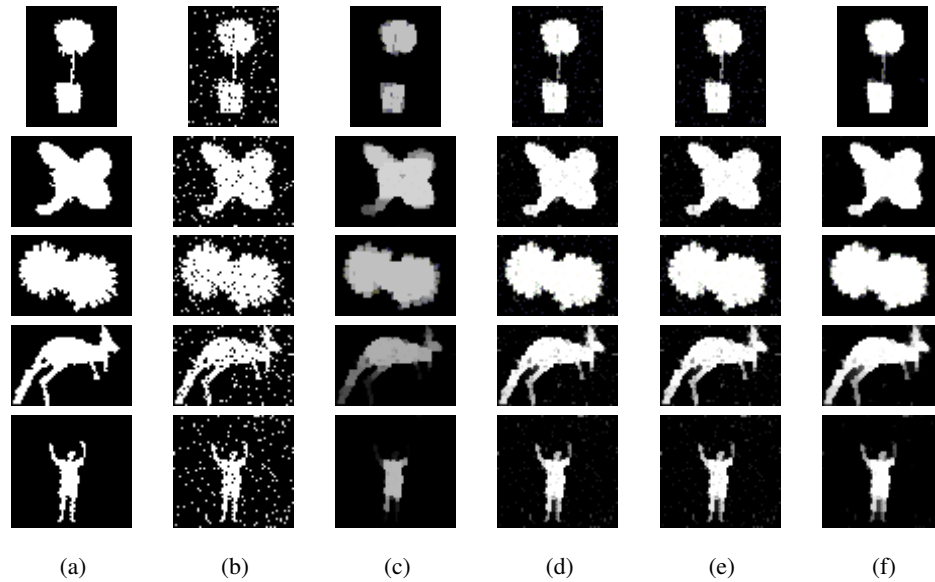


Figure 7: Expected Segmentations. (a) Ground Truth (b) Noised Input (c) Default (0,0,0) (d) P&M (e) Skeleton P&M (f) Slicing Method

Keshet, Joseph, McAllester, David, and Hazan, Tamir. Pac-bayesian approach for minimization of phoneme error rate. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pp. 2224–2227. IEEE, 2011.

Papandreou, G. and Yuille, A. Perturb-and-MAP Random Fields: Using Discrete Optimization to Learn and Sample from Energy Models. In *ICCV*, pp. 193–200, Barcelona, Spain, November 2011. doi: 10.1109/ICCV.2011.6126242.

Porta, Josep M, Vlassis, Nikos, Spaan, Matthijs TJ, and Poupart, Pascal. Point-based value iteration for continuous pomdps. *The Journal of Machine Learning Research*, 7:2329–2367, 2006.

Premachandran, Vittal, Tarlow, Daniel, and Batra, Dhruv. Empirical minimum bayes risk prediction: How to extract an extra few% performance from vision models with just three more parameters. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pp. 1043–1050. IEEE, 2014.

Rother, Carsten, Kolmogorov, Vladimir, and Blake, Andrew. Grabcut: Interactive foreground extraction using iterated graph cuts. In *ACM Transactions on Graphics (TOG)*, volume 23, pp. 309–314. ACM, 2004.

Schmidt, Uwe, Gao, Qi, and Roth, Stefan. A generative perspective on mrfs in low-level vision. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 1751–1758. IEEE, 2010.

Tarlow, Daniel, Adams, Ryan Prescott, and Zemel, Richard S. Randomized Optimum Models for Structured Prediction. In *AISTATS*, pp. 21–23, 2012.

Taskar, Ben, Guestrin, Carlos, and Koller, Daphne. Max-margin markov networks. In *Advances in Neural Information Processing Systems*, pp. None, 2003.

Tsochantaridis, Ioannis, Joachims, Thorsten, Hofmann, Thomas, and Altun, Yasemin. Large margin methods for structured and interdependent output variables. In *Journal of Machine Learning Research*, pp. 1453–1484, 2005.

Williams, Ronald J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

---

# Population Empirical Bayes

---

**Alp Kucukelbir**

Data Science Institute & Computer Science  
Columbia University  
New York, NY 10027

**David M. Blei**

Data Science Institute & Computer Science & Statistics  
Columbia University  
New York, NY 10027

## Abstract

Bayesian predictive inference analyzes a dataset to make predictions about new observations. When a model does not match the data, predictive accuracy suffers. We develop population empirical Bayes (POP-EB), a hierarchical framework that explicitly models the empirical population distribution as part of Bayesian analysis. We introduce a new concept, the latent dataset, as a hierarchical variable and set the empirical population as its prior. This leads to a new predictive density that mitigates model mismatch. We efficiently apply this method to complex models by proposing a stochastic variational inference algorithm, called bumping variational inference (BUMP-VI). We demonstrate improved predictive accuracy over classical Bayesian inference in three models: a linear regression model of health data, a Bayesian mixture model of natural images, and a latent Dirichlet allocation topic model of scientific documents.

## 1 INTRODUCTION

Bayesian modeling is a powerful framework for analyzing structured data. It covers many important methods in probabilistic machine learning, such as regression, mixture models, hidden Markov models, and probabilistic topic models (Bishop, 2006; Murphy, 2012). Bayesian models provide an intuitive language to express assumptions about data, as well as general-purpose algorithms (such as variational methods) to reason under those assumptions.

Bayesian models describe data  $\mathbf{X}$  as a structured probability density with latent variables  $\theta$ ,  $p(\mathbf{X}, \theta) = p(\mathbf{X} | \theta)p(\theta)$ . The first term is the likelihood, the second is the prior. We use this joint density to both analyze data and form predictions. We analyze data through the posterior density of the latent variables,  $p(\theta | \mathbf{X})$ . This conditional density derives from the joint. We form predictions by combining the

likelihood and posterior density as

$$p(\mathbf{x}_{\text{new}} | \mathbf{X}) = \int p(\mathbf{x}_{\text{new}} | \theta)p(\theta | \mathbf{X}) d\theta. \quad (1)$$

This is the Bayesian predictive density, the conditional density of a new observation given the dataset.

Frequentist statistics take a different perspective. Here we analyze a set of observations  $\mathbf{X}$  to draw conclusions about the mechanism  $F$  that gave rise to them.  $F(\mathbf{X})$  is an unknown distribution; it is called the *population*. Nonparametric frequentist methods, like the bootstrap (Efron and Tibshirani, 1994), work directly with  $F$  while still respecting its unknown nature. This leads to powerful tools that work well across many statistical settings. One goal is to use a given dataset to learn about  $F$  and predict new observations; this is predictive inference (Young and Smith, 2005).

**Main idea.** We combine the flexibility of Bayesian modeling with the robustness of nonparametric frequentist statistics. The issue is that Bayesian theory, under frequentist scrutiny, assumes the model is correct (Bernardo and Smith, 2000). But this is rarely true; the model is almost always mismatched, which can lead to brittle data analysis and poor predictions. Our goal is to use the unknown population  $F$  to improve a Bayesian model's predictive performance.

We call our framework population empirical Bayes (POP-EB). It describes a simple procedure. The input is a model  $p(\mathbf{X}, \theta)$  and a dataset  $\mathbf{X}$  of  $N$  observations. For example, a mixture of Gaussians and a dataset of natural images.

1. Draw  $B$  bootstrap samples of the dataset,  $\{\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(B)}\}$ . Each sample is a dataset of size  $N$ , drawn with replacement from the original dataset (Efron and Tibshirani, 1994).
2. Compute the posterior for each bootstrapped dataset,  $p(\theta | \mathbf{X}^{(b)})$ . Evaluate the average predictive accuracy of the original dataset with Equation (1).
3. Pick the bootstrapped dataset  $\mathbf{X}^{(b^*)}$  that best predicts the original dataset. Use the corresponding predictive density  $p(\mathbf{x}_{\text{new}} | \mathbf{X}^{(b^*)})$  to form future predictions.

POP-EB applies to any Bayesian model and can improve its predictive performance. Above we describe its simplest form. Alternatives, which we discuss below, include one that weights the bootstrapped datasets and another that embeds the population into a stochastic variational inference algorithm (Hoffman et al., 2013).

In this paper, we develop, motivate, and study POP-EB. We show that it yields a predictive density that incorporates the unknown population distribution  $F$  in a type of empirical Bayes model (Robbins, 1955, 1964). Compared to traditional Bayesian inference, it better predicts held-out data for models such as regression (Table 1), mixtures of Gaussians (Figure 5), and probabilistic topic models (Figure 7).

**Related work.** There are several running themes in this paper. The first is Bayesian/frequentist compromise and empirical Bayes. A rich literature relates Bayesian and frequentist ideas. Bayarri and Berger (2004) and Aitkin (2010) give thorough reviews. One thread of ideas around combining these two schools of thought is empirical Bayes (EB) (Robbins, 1955; Morris, 1983; Carlin and Louis, 2000). Loosely, EB uses frequentist statistics to estimate prior specifications in Bayesian models. EB enjoys good statistical properties (Efron, 2010); it can explain challenging concepts, such as the James-Stein paradox (Berger, 1985).

The second theme is predictive inference. One approach to Bayesian inference is to study the Bayesian predictive density. The goal is to design models such that the predictive density is high for new observations (Geisser, 1993). Machine learning also strives to develop models that deliver high predictive accuracy (Bishop, 2006). Our method explicitly optimizes the Bayesian predictive density.

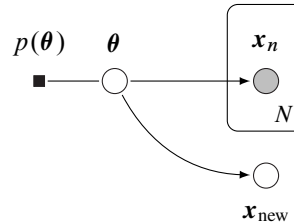
A final theme is model misspecification. To paraphrase Box and Draper (1987), the challenge of Bayesian statistics is that while many models are useful, all of them are wrong. Robust statistics offer some remedies (Berger, 1994), such as using likelihoods and priors that are “insensitive to small deviations from the assumptions” (Huber and Ronchetti, 2009). Our work uses empirical Bayes to induce a model that is robust to misspecification.

## 2 POPULATION EMPIRICAL BAYES

Population empirical Bayes (POP-EB) incorporates the model-independent population  $F$  into Bayesian analysis. We first develop the structure of our framework and then discuss the motivation behind it in Section 2.3.

### 2.1 EMPIRICAL BAYES

Let  $\mathbf{X} = \{\mathbf{x}_n\}_1^N$  be a dataset with  $N$  observations. The dataset is a sample from an unknown population distribution  $F$ . The population is the “true” distribution of the data; it is independent of any model (Shao, 2003).



**Figure 1:** Graphical model for Bayesian predictive inference. The likelihood relates the dataset  $\mathbf{X} = \{\mathbf{x}_n\}_1^N$ , along with the new observation  $\mathbf{x}_{\text{new}}$ , to the latent variables  $\theta$ . Conditioning on the observations and marginalizing over  $\theta$  gives the Bayesian predictive density of Equation (1).

A Bayesian model has two parts. The first is the likelihood,  $p(\mathbf{x}_n | \theta)$ . It relates an observation  $\mathbf{x}_n$  to a set of latent random variables  $\theta$ . If the observations are independent and identically distributed, the likelihood of the dataset becomes  $p(\mathbf{X} | \theta) = \prod_{n=1}^N p(\mathbf{x}_n | \theta)$ .

The second is the prior density,  $p(\theta)$ . This induces the joint density  $p(\mathbf{X}, \theta) = p(\mathbf{X} | \theta) p(\theta)$ . In predictive inference, we additionally consider a new observation  $\mathbf{x}_{\text{new}}$ . It shares the same likelihood of the data,  $p(\mathbf{x}_{\text{new}} | \theta)$ . This expands the joint density to  $p(\mathbf{x}_{\text{new}}, \mathbf{X}, \theta)$ , shown in Figure 1.

A predictive density describes  $\mathbf{x}_{\text{new}}$  given the observed dataset  $\mathbf{X}$ . A simple recipe supplies such a density: condition on the observations and marginalize over the latent variables. This gives the Bayesian predictive density in Equation (1). It depends on the Bayesian posterior density,

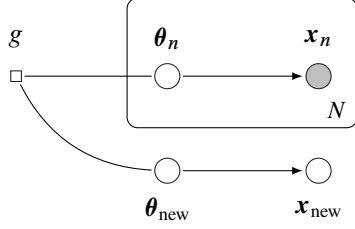
$$p(\theta | \mathbf{X}) = \frac{p(\mathbf{X} | \theta) p(\theta)}{\int p(\mathbf{X} | \theta') p(\theta') d\theta'}, \quad (2)$$

which describes how the latent variables  $\theta$  vary conditioned on a given dataset  $\mathbf{X}$ .

The main idea behind Empirical Bayes (EB) is to build estimates from the population into Bayesian inference (Robbins, 1955; Morris, 1983). EB uses the observed data to estimate parts of a Bayesian model. There are many variants of EB. Robbins (1955) proposed the following approach.

First, augment the model such that each observation  $\mathbf{x}_n$  gets its own set of latent variables  $\theta_n$ . This is sometimes called the “compound sampling model” (Berger, 1985). It has connections to robust inference (Berger and Berliner, 1986) through an intuitive justification.<sup>1</sup> Then, assume the latent variables are exchangeable and distributed according to an unknown prior density  $g$ . Figure 2 shows the EB framework.

<sup>1</sup>When pondering outliers in Bayesian inference, de Finetti explains: “We know that each observation is taken using an instrument with [some] error, but each time chosen at random from a collection of instruments of different precisions” (de Finetti, 1961)



**Figure 2:** Graphical model for empirical Bayes (EB) predictive inference. We augment the Bayesian model and estimate the prior  $g$  from the dataset.

“Nonparametric EB” estimates the prior  $g$  using nonparametric statistics of the data (Robbins, 1955, 1964). The name emphasizes its model-independent approach. “Parametric EB” assumes a parametric family for  $g$  and estimates its parameters from the data (Morris, 1983). The result is a hierarchical Bayesian model whose top-level parameters are determined by the observations. The name emphasizes its model-based approach.

Both variants of EB introduce population information from the dataset  $\mathbf{X}$ . But neither directly builds the population  $F$  into the analysis. How can we adapt EB to directly model the population distribution of data?

## 2.2 POPULATION EMPIRICAL BAYES

We take a two-step approach. We first introduce an additional latent variable into EB and then use it to define a conditional density on  $\theta$ .

The new variable  $\mathbf{Z}$  is a latent dataset. This is a hypothetical dataset that we do not observe. It has the same size and dimension as the observed dataset  $\mathbf{X}$ , but with unknown observations. It lives one level above the  $\theta$  variables.

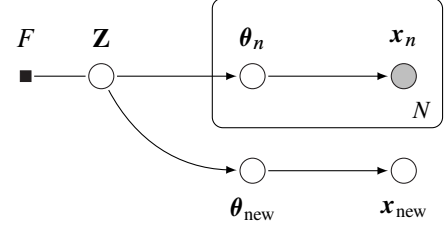
Given the latent dataset, we then define a conditional density on  $\theta$ . We choose a density that depends on the latent dataset,  $p(\theta | \mathbf{Z})$ . Figure 3 shows the framework.

This is a valid EB model; we simply propose a conditional density on  $\theta$  via a new latent variable  $\mathbf{Z}$ . However, it is incomplete. We must also choose a prior on  $\mathbf{Z}$  and then define the form of  $p(\theta | \mathbf{Z})$ .

We set the prior on the latent dataset  $\mathbf{Z}$  to be the population  $F$ . The unknown, model-independent distribution of data acts as the prior on the latent dataset  $\mathbf{Z}$ .

We match the conditional density on  $\theta$  to the Bayesian posterior density of the original Bayesian model. Instead of conditioning on the observed dataset  $\mathbf{X}$ , we condition on the latent dataset  $\mathbf{Z}$ . This is how we interface the frequentist population distribution with the Bayesian model.

This fully describes the POP-EB framework. (We motivate these choices in Section 2.3.)



**Figure 3:** Graphical model for the population empirical Bayes (POP-EB) framework. We introduce  $\mathbf{Z}$ , the latent dataset, and assign the population distribution  $F$  as its prior.

**Definition.** POP-EB defines the joint density

$$p(\mathbf{x}_{\text{new}}, \theta_{\text{new}}, \mathbf{X}, \theta, \mathbf{Z}) = p(\mathbf{x}_{\text{new}} | \theta_{\text{new}}) p(\theta_{\text{new}} | \mathbf{Z}) \times \prod_{n=1}^N p(\mathbf{x}_n | \theta_n) p(\theta_n | \mathbf{Z}) \times F(\mathbf{Z}). \quad (3)$$

To obtain a predictive density from the joint density, we follow the same recipe as before: condition on the observations  $\mathbf{X}$  and marginalize over the latent variables.

Marginalizing over  $\theta$  is straightforward. In addition, we marginalize over the latent dataset  $\mathbf{Z}$ , which gives the predictive density

$$p(\mathbf{x}_{\text{new}} | \mathbf{X}) = \int p(\mathbf{x}_{\text{new}} | \mathbf{Z}) p(\mathbf{Z} | \mathbf{X}) d\mathbf{Z}. \quad (4)$$

This is the POP-EB predictive density. It integrates the Bayesian predictive density over the latent dataset  $\mathbf{Z}$  using the conditional density

$$p(\mathbf{Z} | \mathbf{X}) = \frac{p(\mathbf{X} | \mathbf{Z}) F(\mathbf{Z})}{\int p(\mathbf{X} | \mathbf{Z}') F(\mathbf{Z}') d\mathbf{Z}'}. \quad (5)$$

This is a key conditional density in POP-EB. It describes how the latent dataset varies given the observed dataset. The original Bayesian model prescribes the form of  $p(\mathbf{X} | \mathbf{Z})$  and the population  $F(\mathbf{Z})$  factors in as the prior.

Thus, POP-EB directly incorporates the population  $F$  as a prior on the latent dataset. But the population is, by definition, unknown. We address this next.

**Plug-in principle.** The empirical population  $\hat{F}$  is the distribution that puts weight  $1/N$  on each observation in the observed dataset  $\{\mathbf{x}_n\}_1^N$ . The plug-in estimator of a function of  $F$  is simply the same function evaluated with  $\hat{F}$  instead. In the absence of any other information about the population, the plug-in principle is asymptotically efficient (Efron and Tibshirani, 1994).

The plug-in principle provides a nonparametric estimate of  $F$ . It enjoys a tight connection to the bootstrap and related techniques. We discuss computation in greater detail in Section 3. But now, we owe the reader an explanation.



## 2.3 MOTIVATION

Here is the story so far. The population  $F$  is the model-independent mechanism that generates  $\mathbf{X}$ . Bayesian analysis employs a model  $p(\mathbf{X}, \theta)$ . The model is helpful; it gives strength to the statistical analysis (Young and Smith, 2005). However, Bayesian theory assumes the model is correct.<sup>2</sup> This is not always true; the model is often misspecified.

We focus on predictive inference. Our goal is to help a Bayesian model provide the best predictive accuracy, in spite of model misspecification. So, we seek a way to incorporate the model-independent population  $F$  into our model-based Bayesian analysis.

This is why we set  $F$  as the prior on the latent dataset  $\mathbf{Z}$ . In principle, there is no obstacle in using any other prior density. The prior on  $\mathbf{Z}$  captures knowledge about the data generating mechanism that might otherwise be difficult to express in the model. In our case, this knowledge is precisely the population distribution  $F$ .

Consistency motivates our design of the conditional density on  $\theta$ . Any density that depends on  $\mathbf{Z}$  would be valid. We choose the Bayesian posterior density to mimic the original Bayesian model. Consider the Bayesian predictive density from Equation (1). It integrates the likelihood over the posterior density of the latent variables. The POP-EB predictive density of Equation (4) mirrors this form by integrating the Bayesian predictive density over the posterior density of the latent dataset.

## 3 COMPUTATION

We describe two empirical approximations to the POP-EB predictive density, develop some insight through simulation, and study a linear regression model with real data. More results follow in the empirical study of Section 5.

### 3.1 BOOTSTRAP

The plug-in principle replaces the population  $F$  with its empirical counterpart  $\hat{F}$ . However, direct computation with  $\hat{F}$  is also challenging. The POP-EB predictive density requires evaluating  $p(\mathbf{Z} | \mathbf{X})$ . This involves considering all possible datasets in the support of  $\hat{F}$ , an intractable task.

The bootstrap is a computational technique for approximating functions of  $\hat{F}$ . Define the bootstrapped dataset as  $\mathbf{Z}^{(b)} = \{\mathbf{x}_n^{(b)}\}_1^N$  where each  $\mathbf{x}_n^{(b)}$  is uniformly sampled from  $\hat{F}$  with replacement. The bootstrapped dataset  $\mathbf{Z}^{(b)}$  contains as many observations as  $\mathbf{X}$ ; some observations appear multiple times, others not at all.

All bootstrapped datasets are equally likely. So we can approximate calculations of  $\hat{F}$  with a uniform distribution.

<sup>2</sup>Bernardo and Smith (2000) identify this as the  $\mathcal{M}$ -closed view.

Call this distribution  $\hat{G}$ ; it is a discrete uniform distribution over the  $B$  bootstrapped datasets  $\{\mathbf{Z}^{(b)}\}_1^B$ . This reduces the space of possible datasets to  $\mathcal{O}(B)$ .

### 3.2 MAP APPROXIMATION

A simple way to approximate the POP-EB predictive density is maximum a posteriori (MAP) estimation. In our setup, we find the most likely latent dataset  $\mathbf{Z}^*$  from  $p(\mathbf{Z} | \mathbf{X})$  and plug it into the Bayesian predictive density.

The POP-EB MAP predictive density is simply

$$p_{\text{MAP}}(\mathbf{x}_{\text{new}} | \mathbf{X}) = p(\mathbf{x}_{\text{new}} | \mathbf{Z}^*),$$

where the most likely dataset is

$$\begin{aligned} \mathbf{Z}^* &= \arg \max_{\mathbf{Z}} p(\mathbf{Z} | \mathbf{X}) \\ &= \arg \max_{\mathbf{Z}} p(\mathbf{X} | \mathbf{Z}) \hat{F}(\mathbf{Z}). \end{aligned}$$

MAP estimation evades the intractable denominator in Equation (5), as the maximization only depends on  $\mathbf{Z}$ . However, the maximization is still intractable, so we replace  $\hat{F}$  with  $\hat{G}$  to get

$$\mathbf{Z}^{(b^*)} \approx \arg \max_{\mathbf{Z}^{(b)} \in \hat{G}} p(\mathbf{X} | \mathbf{Z}^{(b)}).$$

This turns out to be a special case of the bumping technique (Tibshirani and Knight, 1999). Bumping is a bootstrap-based method to fit arbitrary models to a dataset; it finds the fit that minimizes some metric over bootstrapped datasets. Our metric is the average predictive density evaluated on the original dataset. Evaluating the metric on the original dataset guards against overfitting; it includes both held-in and held-out samples.

Enumeration spells the procedure out:

1. Draw  $B$  bootstrapped datasets. This gives a set of datasets  $\{\mathbf{Z}^{(b)}\}_1^B \sim \hat{G}$ .<sup>3</sup>
2. For each bootstrap index  $b = 1, \dots, B$ :  
Compute and evaluate the Bayesian predictive density on the original dataset

$$p(\mathbf{X} | \mathbf{Z}^{(b)}) = \prod_{n=1}^N p(\mathbf{x}_n | \mathbf{Z}^{(b)}).$$

3. Return the bootstrap index  $b^*$  that maximizes the Bayesian predictive density above.

We approximate the POP-EB MAP predictive density as

$$p_{\text{MAP}}(\mathbf{x}_{\text{new}} | \mathbf{X}) \approx \int p(\mathbf{x}_{\text{new}} | \theta_{\text{new}}) p(\theta_{\text{new}} | \mathbf{Z}^{(b^*)}) d\theta_{\text{new}}. \quad (6)$$

<sup>3</sup>Tibshirani and Knight (1999) recommend including the observed dataset. We follow their advice.

It has the same form as the Bayesian predictive density, but integrates over the Bayesian posterior conditioned on the bootstrapped dataset  $\mathbf{Z}^{(b^*)}$ .

### 3.3 FULL BAYESIAN APPROXIMATION

We also consider directly evaluating the POP-EB predictive density of Equation (4). In general, the posterior  $p(\mathbf{Z} | \mathbf{X})$  is intractable. Luckily, replacing  $\hat{F}$  with  $\hat{G}$  reduces the integral to a finite sum over  $B$ . We call this a full Bayes (FB) approximation, as it is an exact evaluation of the POP-EB predictive density under the  $\hat{G}$  approximation of the empirical population. (The supplement contains a derivation.)

The POP-EB FB predictive density is a weighted sum,

$$p_{\text{FB}}(\mathbf{x}_{\text{new}} | \mathbf{X}) = \sum_{b=1}^B w_b p(\mathbf{x}_{\text{new}} | \mathbf{Z}^{(b)}),$$

where the weights are

$$w_b = \frac{p(\mathbf{X} | \mathbf{Z}^{(b)}) \hat{G}(\mathbf{Z}^{(b)})}{\sum_b p(\mathbf{X} | \mathbf{Z}^{(b)}) \hat{G}(\mathbf{Z}^{(b)})} = \frac{p(\mathbf{X} | \mathbf{Z}^{(b)})}{\sum_b p(\mathbf{X} | \mathbf{Z}^{(b)})},$$

and the  $\mathbf{Z}^{(b)}$  are drawn from  $\hat{G}$ . The probabilities  $\hat{G}(\mathbf{Z}^{(b)})$  are all equal to  $1/B$ , and so they disappear. As with the MAP case, the intractable denominator from Equation (5) also drops out of the calculation.

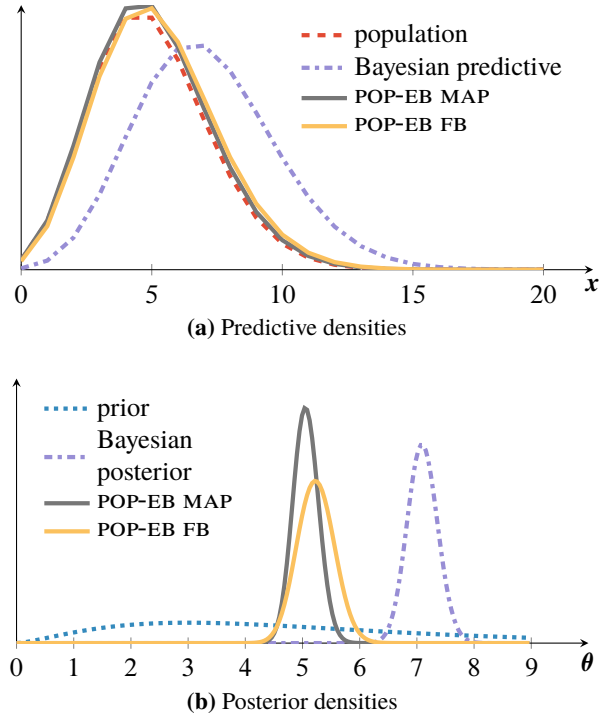
### 3.4 SIMULATION STUDY

To investigate these POP-EB quantities, we construct a toy example of model mismatch. The example is intentionally simple; it provides insight into why and how POP-EB mitigates model misspecification.

Consider that we typically observe data from a Poisson distribution with rate  $\theta = 5$ . (For instance, a router receives packets over a network; the wait-times are the measurements.) We model the data using a Poisson likelihood  $p(\mathbf{x} | \theta) = \text{Poisson}(\theta)$ , and center a Gamma prior at  $\theta = 5$  as  $p(\theta) = \text{Gam}(\alpha = 2.5, \beta = 0.5)$ .

Imagine that five percent of the time, the network fails. During these failures, the wait-times come from a different Poisson with rate  $\theta = 50$ . The population describes a mixture of two Poisson distributions. But the single Poisson model likelihood does not. A good predictive density should accurately describe the population.

Figure 4a shows the predictive densities. The Bayesian predictive density exhibits poor predictive accuracy; it describes neither of the two Poisson distributions in the population. In contrast, both POP-EB predictive densities match the dominant Poisson distribution. POP-EB uses the empirical population to focus on the observations in the dataset that give it greater predictive power.



**Figure 4:** Gamma-Poisson simulation. The population in subpanel (a) has an small extra bump at 50 (not shown).

Figure 4b shows the underlying posterior densities on the latent variable  $\theta$  that describes the wait-time. For POP-EB MAP this is the posterior density  $p(\theta | \mathbf{Z}^{(b^*)})$ ; a single Gamma distribution that, when put through the Bayesian predictive density, gives the highest accuracy. For the POP-EB FB case, the posterior is a weighted sum of Gamma distributions. Both POP-EB posterior densities sit closer to the dominant rate of  $\theta = 5$ . This explains the behavior of the POP-EB predictive densities in Figure 4a.

The POP-EB MAP predictive density is easy to compute; it is a negative binomial distribution, like the Bayesian predictive density. In contrast, the POP-EB FB predictive density must be numerically calculated using all bootstrap samples; it has no simple analytic form.

We generate these plots using  $B = 100$  bootstrapped datasets. The exact shape of the POP-EB densities depend on the bootstrapped datasets, but they are reproducibly closer to  $\theta = 5$ . The results are also insensitive to different prior configurations, such as a sharp prior centered at  $\theta = 5$ . EB is also of little use here; it estimates a nearly flat prior on  $\theta$  from the data. (See supplementary note and code.)

### 3.5 BAYESIAN LINEAR REGRESSION

We now apply the POP-EB framework to a real-world dataset. Consider a Bayesian linear regression model. The likelihood is a Gaussian distribution and the latent random variables are the regression coefficients  $\beta$  and the variance  $\sigma^2$ . Using

**Table 1:** Bayesian linear regression predictive accuracy results on random held-out splits of the bodyfat dataset.

|            | Split #1    |               |               | Split #2    |               |               |
|------------|-------------|---------------|---------------|-------------|---------------|---------------|
|            | logp        | MSE           | MAE           | logp        | MSE           | MAE           |
| Bayes      | 0.67        | 5.2e-3        | 5.7e-2        | 0.83        | 7.6e-3        | 6.6e-2        |
| POP-EB MAP | <b>1.26</b> | 3.2e-3        | 4.3e-2        | 1.18        | 5.1e-3        | 5.4e-2        |
| POP-EB FB  | 1.25        | <b>3.0e-3</b> | <b>4.2e-2</b> | 1.18        | <b>4.4e-3</b> | <b>5.2e-2</b> |
|            | Split #3    |               |               | Split #4    |               |               |
|            | logp        | MSE           | MAE           | logp        | MSE           | MAE           |
| Bayes      | 0.85        | 6.5e-3        | 5.4e-2        | 0.81        | 7.4e-3        | 6.7e-2        |
| POP-EB MAP | <b>1.24</b> | 4.0e-3        | 4.9e-2        | <b>1.21</b> | 5.1e-3        | 5.5e-2        |
| POP-EB FB  | 1.23        | <b>3.3e-3</b> | <b>4.4e-2</b> | 1.19        | <b>4.2e-3</b> | <b>5.3e-2</b> |
|            | Split #5    |               |               | Split #6    |               |               |
|            | logp        | MSE           | MAE           | logp        | MSE           | MAE           |
| Bayes      | 0.82        | 7.2e-3        | 6.9e-2        | 0.44        | 3.1e-2        | 9.2e-2        |
| POP-EB MAP | <b>1.15</b> | 6.2e-3        | 6.1e-2        | 0.75        | 1.7e-2        | 7.1e-2        |
| POP-EB FB  | 1.14        | <b>4.9e-3</b> | <b>5.2e-2</b> | <b>0.76</b> | <b>1.6e-2</b> | <b>6.6e-2</b> |

conjugate priors (Gaussian for the coefficients and inverse Gamma for the variance) gives a Bayesian predictive density that follows a t-distribution (Murphy, 2012). We posit an uninformative prior.

We study the predictive performance of both POP-EB predictive densities on the bodyfat dataset (StatLib, 1995). Accurate measurements of body fat are costly. The dataset contains the body fat percentages of  $N = 252$  men along with 14 other features that are cheaper to measure. We want to predict body fat percentage using these 14 features.

We randomly extract datasets of 200 measurements and hold the remaining 52 to evaluate predictive accuracy. Table 1 reports the average logarithm of the predictive densities (logp), along with mean squared error (MSE) and mean absolute error (MAE). In all cases, the POP-EB predictive densities reach higher predictive accuracy on held-out data. The POP-EB FB density performs similarly to the POP-EB MAP density, but exhibits slightly better MSE and MAE values. Since the dataset is small, we split it six times. These results are reproducible across a variety of splits. We use  $B = 25$  bootstrap samples. (See supplementary code.)

The POP-EB FB density is a better approximation to the POP-EB predictive density than its MAP counterpart. The results in Table 1 corroborate this. However, POP-EB FB density lack an analytic form. In contrast, the POP-EB MAP predictive density offers an attractive improvement in predictive accuracy while maintaining the form of the Bayesian predictive density. We now turn to approximating it for complex Bayesian models.

## 4 POPULATION EMPIRICAL VARIATIONAL INFERENCE

Modern Bayesian statistics and machine learning has moved well past simple conjugate models like Bayesian linear re-

gression. In complex models, such as Bayesian mixture models (Bishop, 2006) or probabilistic topic models (Blei et al., 2003), the posterior and predictive densities are intractable to compute. Monte Carlo sampling and variational techniques are two popular frameworks for approximation.

In this section, we develop an efficient variational approximation to the POP-EB MAP predictive density.

### 4.1 VARIATIONAL INFERENCE

Variational inference is an optimization-based approach to approximate posterior computation in a Bayesian model  $p(\mathbf{X}, \theta)$  (Jordan et al., 1999; Wainwright and Jordan, 2008). The idea is to posit a simple parameterized density family over the latent variables,  $q(\theta; \lambda)$ . We then seek the member of the family that is closest in Kullback-Leibler (KL) divergence to the true posterior density  $p(\theta | \mathbf{X})$ . Minimizing  $\text{KL}(q || p)$  is equivalent to maximizing a lower bound on the marginal density of the data. This gives a variational objective function, called the evidence lower bound (ELBO)

$$\mathcal{L}(\mathbf{X}, \lambda) = \mathbb{E}_q [\log p(\mathbf{X}, \theta)] - \mathbb{E}_q [\log q(\theta; \lambda)].$$

Variational inference maximizes this objective function with respect to the set of parameters  $\lambda$ .

In mean-field variational inference, we assume the variational density fully factorizes. This divides the variational parameters into  $M$  parts,  $\lambda = \{\lambda_m\}_1^M$ . We then maximize the ELBO using coordinate ascent (Jordan et al., 1999). This means iteratively maximizing one variational parameter at a time, holding all others fixed. The separation of latent variables leads to independent updates for each variational parameter. This coordinate ascent scheme guarantees convergence to a local maximum of the ELBO (Bishop, 2006).

### 4.2 APPROXIMATING THE POP-EB MAP PREDICTIVE DENSITY

Our aim is to employ variational inference to approximate the POP-EB MAP predictive density. It shares the same form as the Bayesian predictive density, but integrates over the Bayesian posterior density evaluated on a particular bootstrapped dataset. To that end, consider the ELBO evaluated on a bootstrapped dataset,  $\mathcal{L}(\mathbf{Z}^{(b)}, \lambda)$ .

The variational objective becomes a joint optimization of the variational parameters and the bootstrap index

$$\lambda_{\dagger}^{(b^*)} = \arg \max_{\lambda} \mathcal{L}(\mathbf{Z}^{(b^*)}, \lambda)$$

$$b^* = \arg \max_b \prod_{n=1}^N \int p(x_n | \theta_n) q(\theta_n; \lambda_{\dagger}^{(b)}) d\theta_n.$$

The two optimization problems are coupled: the first seeks the best approximation to the Bayesian posterior density while the second seeks the latent dataset index that gives the highest predictive accuracy. The variational density  $q(\theta; \lambda_{\dagger}^{(b^*)})$  is the closest KL approximation to the Bayesian posterior density inside the POP-EB MAP predictive density of Equation (6).

The naïve way to solve the joint optimization above is to adapt our earlier technique from Section 3.2. Bootstrap the dataset  $B$  times, maximize the ELBO for each dataset, and choose the one that gives the best predictive performance. This is a costly procedure. It requires multiple maximizations of the ELBO, which we wish to avoid.

### 4.3 BUMPING VARIATIONAL INFERENCE

We propose a stochastic optimization algorithm that maximizes the ELBO once. We weave bumping into each iteration of the optimization. We call this method bumping variational inference (BUMP-VI) (Algorithm 1). At a high level, the algorithm works as follows.

Consider a single iteration. We bootstrap the dataset  $B$  times. For each bootstrapped dataset, we compute a gradient  $\mathbf{g}_{\lambda}^{(b)}$  in the parameters  $\lambda$ . These gradients are noisy estimates of the “true” gradient, had we taken the naïve approach above and determined which bootstrapped dataset  $\mathbf{Z}^{(b^*)}$  led to the best predictive performance. The cost of computing  $B$  gradients is small in many Bayesian models. (The supplement describes an efficient implementation.)

Then, we employ the bumping procedure from Section 3.2. This means taking a step in the parameters for each bootstrapped dataset and evaluating the Bayesian predictive density on the original dataset. This evaluation is the main added cost of BUMP-VI over classical variational methods. We pick the index  $b^{(*)}$  that gives the highest predictive performance and take a step in the direction it indexes.

BUMP-VI is a stochastic optimization method; the step-size sequence matters for establishing convergence guarantees (Robbins and Monro, 1951). We use a constant step-size as it isolates the performance of the algorithm from any sequence-related effects and provides quantifiable error bounds (Nemirovski et al., 2009). Code is available at <https://github.com/Blei-Lab/lda-bump-cpp>.

## 5 EMPIRICAL STUDY

We apply BUMP-VI to two complex tasks: Bayesian mixture modeling of image histograms and latent Dirichlet allocation (LDA) topic modeling of a scientific text corpus. We compare BUMP-VI to coordinate ascent variational inference. In both examples, BUMP-VI uniformly reaches higher predictive accuracy. We first present both sets of results and then discuss performance.

---

### Algorithm 1: Bumping Variational Inference

---

**Input:** Model  $p(\mathbf{X}, \theta)$ , variational family  $q(\theta; \lambda)$   
**Result:** Optimized  $q(\theta; \lambda_{\dagger}^{(b^*)})$  approximation

Choose the number of bootstraps  $B$   
 Choose a step-size sequence  $\{\rho_{(i)}\}_1^{\infty}$   
 Initialize parameters  $\lambda_{(0)}$ , iteration  $i = 0$

**while**  $\|\lambda_{(i+1)} - \lambda_{(i)}\|$  is above some threshold **do**

**for**  $b = 1$  **to**  $B$  **do**

    Draw a bootstrap sample  $\mathbf{Z}^{(b)} \sim \hat{G}$

    Calculate gradient of parameters  $\lambda$

$$\mathbf{g}_{\lambda}^{(b)} \leftarrow \nabla_{\lambda} \mathcal{L}(\mathbf{Z}^{(b)}, \lambda)$$

**end**

  Choose the bootstrap index that maximizes the Bayesian predictive density on the dataset

$$b^* \leftarrow \arg \max_b \prod_{n=1}^N \int p(x_n | \theta_n) q(\theta_n; \lambda_{(i)} + \rho_{(i)} \mathbf{g}_{\lambda}^{(b)}) d\theta_n$$

  Take a step in direction indexed by  $b^*$

$$\lambda_{(i+1)} \leftarrow \lambda_{(i)} + \rho_{(i)} \mathbf{g}_{\lambda}^{(b^*)}$$

  Update iteration counter

$$i \leftarrow i + 1$$

**end**

Return variational parameters

$$\lambda_{\dagger}^{(b^*)} \leftarrow \lambda_{(i+1)}$$

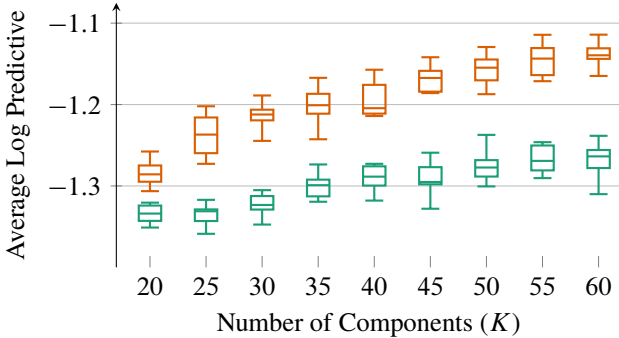

---

### 5.1 BAYESIAN MIXTURE MODEL

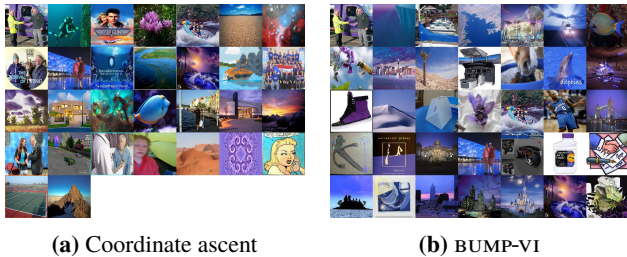
Consider a Gaussian mixture model (GMM) with a Gaussian-Gamma prior on the mixture component means and precisions, and a Dirichlet prior on the proportions. These are conditionally conjugate priors that lead to straightforward coordinate ascent update equations (Bishop, 2006).

The imageCLEF dataset has 576-dimensional color histograms of natural images (Villegas et al., 2013). We randomly select 5 000 images as our dataset and choose another 1 000 images for predictive accuracy tests. We standardize the mean and variance of the set of histograms. Thus, the hyperparameters for the Gaussian-Gamma prior are zero on the component means and one on the precisions. The hyperparameter on the Dirichlet is  $1/K$  where  $K$  is the number of components. We set the number of bootstrap samples to  $B = 10$  and the step-size to  $\rho = 0.1$ .

We compare coordinate ascent to BUMP-VI across a range of components. Figure 5 displays these results. BUMP-VI attains a higher average log predictive evaluated on the held-out set. We run each algorithm ten times per configuration to account for random initialization of the initial parameters.



**Figure 5:** Average log-predictive density of 1 000 held-out images. BUMP-VI (orange) reaches a higher predictive accuracy than coordinate ascent (green) across a range of configurations. Boxplots show the median, quartiles, and 1.5 interquartile range of ten repeats per configuration.



**Figure 6:** The “blue-purple” GMM ( $K = 40$ ) coordinate ascent component has some extraneous images. The bumping component is qualitatively more uniform.

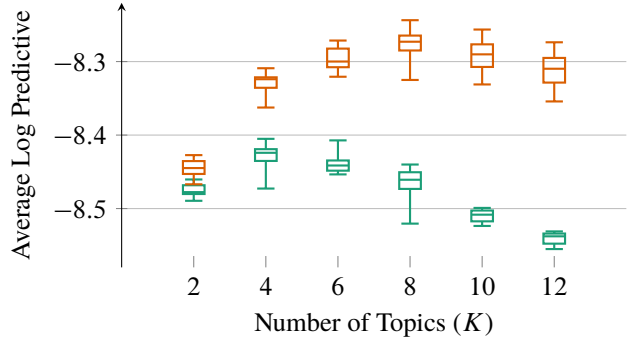
Both algorithms converge in fewer than 75 iterations.

Figure 6 presents the effect of reaching higher predictive accuracy. We assign each image to its most probable mixture component and pick the one with “blue-purple” images. Though this is a subjective matter, the BUMP-VI component appears more uniform than its counterpart. The coordinate ascent component seems to have red-toned images that might belong in a different component.

## 5.2 LATENT DIRICHLET ALLOCATION

We study LDA with a corpus of 5 000 randomly selected abstracts from the arXiv repository. The abstracts are short ( $\sim 150$  words) and the vocabulary is large ( $\sim 12\,000$  unique words). The arXiv exhibits jargon-heavy abstracts, which leads to a large vocabulary. This makes for a challenging dataset; because of the vocabulary size, we only expect to identify a few topics in a corpus of size 5 000.

LDA has two Dirichlet priors. We set the hyperparameter on the topics distributions to be  $1/K$  where  $K$  is the number of topics. The hyperparameter on the topics is 0.005, which is approximately  $60/V$  where  $V$  is the vocabulary size. We set the number of bootstrap samples to  $B = 10$  and the



**Figure 7:** Average per-word log-predictive density of 1 000 held-out abstracts. BUMP-VI (orange) reaches a higher predictive accuracy than coordinate ascent (green) across a range of model configurations.

step-size to  $\rho = 0.05$ .

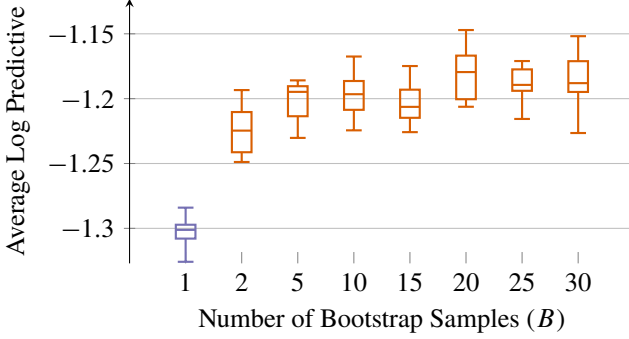
We study BUMP-VI across a range of topics. Figure 7 plots a comparison to coordinate ascent. BUMP-VI attains a higher average per-word log predictive evaluated on the held-out set. Tables 3 and 4 show how this difference affects the topics. While six of the topics are similar, BUMP-VI finds two topics with higher predictive power. As with mixture components, examining topics is a subjective activity. Both algorithms converged in fewer than 150 iterations.

## 5.3 MITIGATING MODEL MISMATCH

There is no reason to believe that natural images are truly distributed as a mixture of Gaussians. The same holds for the “bag of words” assumption that underlies LDA. With real data, there is always some level of inherent model mismatch. BUMP-VI attempts to mitigate this effect.

The performance gap widens with the number of components. We might expect a simpler model to be more severely mismatched. This demonstrates that POP-EB cannot rectify the choice of an inappropriate model. For example, two topics are simply too few to accurately model our arXiv corpus; both algorithms do poorly. However, with twelve topics, coordinate ascent does even worse. This may be due to other effects, such as local maxima in the variational objective function (Wainwright and Jordan, 2008).

Does BUMP-VI outperform classical techniques because it reaches a better local maximum of the ELBO? The answer is no. Evaluating the ELBO on the observed dataset gives similar numerical values for both methods. (In our experiments, coordinate ascent usually reaches a slightly higher number than BUMP-VI.) This is not surprising, as BUMP-VI solves a different optimization problem than coordinate ascent variational inference; it approximates the POP-EB MAP predictive density instead of the Bayesian predictive density.



**Figure 8:** BUMP-VI sensitivity study with the GMM ( $K = 40$ ) model. Average log-predictive density of 1 000 held-out images, over ten repeats.

**Table 2:** Runtime ratios of BUMP-VI compared to coordinate ascent. An efficient re-weighting strategy gives ratios that are less than  $B$ . (See supplementary note.)

| $B$     | 2    | 5    | 10   | 15   | 30    |
|---------|------|------|------|------|-------|
| BUMP-VI | 1.8× | 3.2× | 5.6× | 7.7× | 15.1× |

#### 5.4 SENSITIVITY TO BOOTSTRAP SAMPLES AND COMPUTATIONAL COST

We also study the sensitivity of BUMP-VI to the number of bootstrap samples  $B$  (Figure 8). Tibshirani and Knight (1999) recommend using 20-30 bootstrap samples for bumping, but as few as five appear to perform well. This is because BUMP-VI, in a loose sense, re-samples the dataset  $B$  times the number of iterations. For completeness, we also compare the case of  $B = 1$ . This is equivalent to stochastic variational inference (SVI) with “minibatch” size equal to  $N$  (Hoffman et al., 2013). This shows that the performance gain in BUMP-VI is not due to stochastic escapes from local maxima. (Though, it likely benefits from it.)

The POP-EB framework incurs a computational cost proportional to the choice of  $B$ . BUMP-VI reduces this cost in two ways. The first is due to the iterative resampling effect mentioned above. The second is due to an efficient calculation of the  $B$  gradients at each iteration. (See supplementary note.) Table 2 shows that BUMP-VI is less than  $B$  times the cost of coordinate ascent variational inference.

## 6 CONCLUSION

Mismatched models exhibit poor predictive performance. The POP-EB predictive density mitigates this effect by incorporating the population  $F$  into Bayesian analysis. The POP-EB MAP predictive density is attractively simple; it explores the space of bootstrapped dataset to find the one with highest predictive power. BUMP-VI extends this idea to variational

inference; it delivers an efficient algorithm with promising results on real-world datasets.

POP-EB, like EB, uses the dataset twice: once to estimate the prior and again during inference. There are alternatives to consider, such as cross-validation and bias correction (Efron and Tibshirani, 1997; Gelman et al., 2013). Directly modeling the expected predictive performance of future data should also improve POP-EB.

**Table 3:** Coordinate ascent LDA topics ( $K = 8$ ).

| Topic 1      | Topic 2       | Topic 3      | Topic 4    |
|--------------|---------------|--------------|------------|
| charge       | gravitational | et           | knowledge  |
| ground       | dynamical     | al           | often      |
| induced      | cosmological  | accurate     | introduced |
| regime       | physics       | test         | novel      |
| length       | review        | extended     | among      |
| leads        | black         | identified   | research   |
| dependent    | background    | signal       | called     |
| transport    | universe      | period       | easily     |
| fluctuations | gr            | during       | key        |
| scattering   | gravity       | correlation  | processing |
| Topic 5      | Topic 6       | Topic 7      | Topic 8    |
| invariant    | quark         | algorithm    | galaxies   |
| algebra      | cross         | random       | stars      |
| operator     | production    | applications | galaxy     |
| theorem      | heavy         | efficient    | stellar    |
| defined      | qcd           | finally      | gas        |
| g            | collisions    | probability  | galactic   |
| generalized  | predictions   | network      | sources    |
| complex      | corrections   | optimal      | objects    |
| explicit     | experiment    | gaussian     | source     |
| infinite     | photon        | distributed  | sample     |

**Table 4:** BUMP-VI LDA topics ( $K = 8$ ).

| Topic 1      | Topic 2       | Topic 3      | Topic 4     |
|--------------|---------------|--------------|-------------|
| induced      | cosmological  | accurate     | performance |
| charge       | universe      | test         | research    |
| regime       | gravitational | identified   | development |
| electronic   | review        | during       | novel       |
| transport    | cosmic        | better       | key         |
| length       | dark          | mostly       | developed   |
| leads        | gravity       | errors       | processing  |
| temperatures | background    | scales       | called      |
| frequency    | relativity    | though       | knowledge   |
| influence    | physics       | sensitivity  | analyze     |
| Topic 5      | Topic 6       | Topic 7      | Topic 8     |
| algebra      | quark         | algorithm    | galaxies    |
| invariant    | production    | random       | stars       |
| defined      | cross         | probability  | galaxy      |
| operator     | heavy         | network      | stellar     |
| theorem      | qcd           | applications | galactic    |
| g            | collisions    | quant        | gas         |
| complex      | predictions   | complexity   | sample      |
| construct    | momentum      | finally      | sources     |
| conjecture   | photon        | problems     | objects     |
| special      | detector      | optimal      | source      |

#### Acknowledgments

We thank Allison Chaney, Laurent Charlin, Stephan Mandt, Kui Tang, Yixin Wang, and the reviewers for their insightful comments. DMB is supported by NSF IIS-1247664, ONR N00014-11-1-0651, and DARPA FA8750-14-2-0009.

## References

- Aitkin, M. A. (2010). *Statistical Inference: an Integrated Bayesian/Likelihood Approach*. Chapman & Hall/CRC.
- Bayarri, M. J. and Berger, J. O. (2004). The interplay of Bayesian and frequentist analysis. *Statistical Science*, pages 58–80.
- Berger, J. (1994). An overview of robust Bayesian analysis. *Test*, 3(1):5–124.
- Berger, J. and Berliner, L. M. (1986). Robust Bayes and empirical Bayes analysis with  $\epsilon$ -contaminated priors. *The Annals of Statistics*, pages 461–486.
- Berger, J. O. (1985). *Statistical decision theory and Bayesian analysis*. Springer.
- Bernardo, J. M. and Smith, A. F. (2000). *Bayesian Theory*. John Wiley & Sons.
- Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer New York.
- Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1022.
- Box, G. E. and Draper, N. R. (1987). *Empirical Model-building and Response Surfaces*. John Wiley & Sons.
- Carlin, B. and Louis, T. (2000). *Bayes and Empirical Bayes Methods for Data Analysis, Second Edition*. Chapman & Hall/CRC. Taylor & Francis.
- de Finetti, B. (1961). The Bayesian approach to the rejection of outliers. In *Proceedings of the Fourth Berkeley Symposium on Probability and Statistics*, pages 199–210.
- Efron, B. (2010). *Large-scale inference: empirical Bayes methods for estimation, testing, and prediction*. Cambridge University Press.
- Efron, B. and Tibshirani, R. (1997). Improvements on cross-validation: The 632+ bootstrap method. *Journal of the American Statistical Association*, pages 548–560.
- Efron, B. and Tibshirani, R. J. (1994). *An Introduction to the Bootstrap*. CRC Press.
- Geisser, S. (1993). *Predictive Inference*. CRC Press.
- Gelman, A., Hwang, J., and Vehtari, A. (2013). Understanding predictive information criteria for Bayesian models. *Statistics and Computing*, pages 1–20.
- Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. (2013). Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347.
- Huber, P. and Ronchetti, E. (2009). *Robust Statistics*. Wiley.
- Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. (1999). An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233.
- Morris, C. N. (1983). Parametric empirical Bayes inference: theory and applications. *Journal of the American Statistical Association*, 78(381):47–55.
- Murphy, K. P. (2012). *Machine Learning: a Probabilistic Perspective*. MIT Press.
- Nemirovski, A., Juditsky, A., Lan, G., and Shapiro, A. (2009). Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19(4):1574–1609.
- Robbins, H. (1955). An empirical Bayes approach to statistics. In *Proceedings of the Third Berkeley Symposium on Probability and Statistics*, pages 157–164.
- Robbins, H. (1964). The empirical Bayes approach to statistical decision problems. *The Annals of Mathematical Statistics*, pages 1–20.
- Robbins, H. and Monroe, S. (1951). A stochastic approximation method. *The Annals of Mathematical Statistics*, pages 400–407.
- Shao, J. (2003). *Mathematical Statistics*. Springer.
- StatLib (1995). <http://lib.stat.cmu.edu/datasets/bodyfat>.
- Tibshirani, R. and Knight, K. (1999). Model search by bootstrap “bumping”. *Journal of Computational and Graphical Statistics*, 8(4):671–686.
- Villegas, M., Paredes, R., and Thomee, B. (2013). Overview of the ImageCLEF 2013 Scalable Concept Image Annotation Subtask. In *CLEF 2013 Evaluation Labs and Workshop, Online Working Notes*.
- Wainwright, M. J. and Jordan, M. I. (2008). Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305.
- Young, G. A. and Smith, R. L. (2005). *Essentials of Statistical Inference*. Cambridge University Press.

---

# Encoding Markov Logic Networks in Possibilistic Logic

---

**Ondřej Kuželka**

School of CS & Informatics  
Cardiff University  
Cardiff, UK

**Jesse Davis**

Department of Computer Science  
KU Leuven  
Leuven, Belgium

**Steven Schockaert**

School of CS & Informatics  
Cardiff University  
Cardiff, UK

## Abstract

Markov logic uses weighted formulas to compactly encode a probability distribution over possible worlds. Despite the use of logical formulas, Markov logic networks (MLNs) can be difficult to interpret, due to the often counter-intuitive meaning of their weights. To address this issue, we propose a method to construct a possibilistic logic theory that exactly captures what can be derived from a given MLN using maximum a posteriori (MAP) inference. Unfortunately, the size of this theory is exponential in general. We therefore also propose two methods which can derive compact theories that still capture MAP inference, but only for specific types of evidence. These theories can be used, among others, to make explicit the hidden assumptions underlying an MLN or to explain the predictions it makes.

## 1 INTRODUCTION

Markov logic [22] and possibilistic logic [9] are two popular logics for modelling uncertain beliefs. Both logics share a number of important characteristics. At the syntactic level, formulas correspond to pairs  $(\alpha, \lambda)$ , consisting of a classical formula  $\alpha$  and a certainty weight  $\lambda$ , while at the semantic level, sets of these formulas induce a mapping from possible worlds to  $[0, 1]$ , encoding the relative plausibility of each possible world.

Despite their close similarities, however, Markov logic and possibilistic logic have been developed in different communities and for different purposes: Markov logic has mainly been studied in a machine learning context whereas possibilistic logic has been studied as a knowledge representation language. This reflects the complementary strengths and weaknesses of these logics. On the one hand, the qualitative nature of possibilistic logic makes it challenging to use for learning; although a few interesting approaches for

learning possibilistic logic theories from data have been explored (e.g. [24]), their impact on applications to date has been limited. On the other hand, the intuitive meaning of Markov logic theories is often difficult to grasp, which limits the potential of Markov logic for knowledge representation. The main culprit is that the meaning of a theory can often not be understood by looking at the individual formulas in isolation. This issue, among others, has been highlighted in [26], where coherence measures are proposed that evaluate to what extent the formulation of a Markov logic theory is misleading.

**Example 1.** Consider the following Markov logic formulas:

$$\begin{aligned} +\infty : & \quad \text{antarctic-bird}(X) \rightarrow \text{bird}(X) \\ 10 : & \quad \text{bird}(X) \rightarrow \text{flies}(X) \\ 5 : & \quad \text{antarctic-bird}(X) \rightarrow \neg \text{flies}(X) \end{aligned}$$

While the last formula might appear to suggest that antarctic birds cannot fly, in combination with the other two formulas, it merely states that antarctic birds are less likely to fly than birds in general.

Possibilistic logic is based on a purely qualitative, comparative model of uncertainty: while a Markov logic theory compactly encodes a probability distribution over the set of possible worlds, a possibilistic logic theory merely encodes a ranking of these possible worlds. Even though a probability distribution offers a much richer uncertainty model, many applications of Markov logic are based on MAP inference, which only relies on the ranking induced by the probability distribution.

In this paper, we first show how to construct a possibilistic logic theory  $\Theta$ , given a Markov logic theory  $\mathcal{M}$ , such that the conclusions that we can infer from  $\Theta$  are exactly those conclusions that we can obtain from  $\mathcal{M}$  using MAP inference. Our construction can be seen as the syntactic counterpart of the probability-possibility transformation from [10]. In principle, it allows us to combine the best of both worlds, using  $\mathcal{M}$  for making predictions while using  $\Theta$  for elucidating the knowledge that is captured by  $\mathcal{M}$  (e.g. to verify



that the theory  $\mathcal{M}$  is sensible). However, the size of  $\Theta$  can be exponential in the size of  $\mathcal{M}$ , which is unsurprising given that the computational complexity of MAP inference is higher than the complexity of inference in possibilistic logic. To overcome this problem, we begin by studying ground (i.e. propositional) theories and propose two novel approaches for transforming a ground MLN into a compact ground possibilistic logic theory that still correctly captures MAP inference, but only for specific types of evidence (e.g. sets of at most  $k$  literals). Then we lift one of these approaches such that it can transform a first-order MLN into a first-order possibilistic logic theory. Finally, we present several examples that illustrate how the transformation process can be used to help identify unintended consequences of a given MLN, and more generally, to better understand its behaviour.

The remainder of the paper is structured as follows. In the next section, we provide some background on Markov logic and possibilistic logic. In Section 3, we analyse the relation between MAP inference in ground Markov logic networks and possibilistic logic inference, introducing in particular two methods for deriving compact theories. Section 4 then discusses how we can exploit the symmetries in the case of an ungrounded Markov logic network, while Section 5 provides some illustrative examples. Finally, we provide an overview of related work in Section 6.

Due to space limitations, some of the proofs have been omitted from this paper. These proofs can be found in an online appendix.<sup>1</sup>

## 2 BACKGROUND

### 2.1 MARKOV LOGIC

A Markov logic network (MLN) [22] is a set of pairs  $(F, w_F)$ , where  $F$  is a formula in first-order logic and  $w_F$  is a real number, intuitively reflecting a penalty that is applied to possible worlds (i.e. logical interpretations) that violate  $F$ . In examples, we will also use the notation  $w_F : F$  to denote the formula  $(F, w_F)$ . An MLN serves as a template for constructing a propositional Markov network. In particular, given a set of constants  $C$ , an MLN  $\mathcal{M}$  induces the following probability distribution on possible worlds  $\omega$ :

$$p_{\mathcal{M}}(\omega) = \frac{1}{Z} \exp \left( \sum_{(F, w_F) \in \mathcal{M}} w_F n_F(\omega) \right), \quad (1)$$

where  $n_F(x)$  is the number of true groundings of  $F$  in the possible world  $\omega$ , and  $Z$  is a normalization constant to ensure that  $p_{\mathcal{M}}$  can be interpreted as a probability distribution. Sometimes, formulas  $(F, w_F)$  are considered where  $w_F = +\infty$ , to represent hard constraints. In such cases,

we define  $p_{\mathcal{M}}(\omega) = 0$  for all possible worlds that do not satisfy all of the hard constraints, and only formulas with a real-valued weight are considered in (1) for the possible worlds that do. Note that a Markov logic network can be seen as a weighted set of propositional formulas, which are obtained by grounding the formulas in  $\mathcal{M}$  w.r.t. the set of constants  $C$  in the usual way. In the particular case that all formulas in  $\mathcal{M}$  are already grounded,  $\mathcal{M}$  corresponds to a theory in penalty logic [13].

One common inference task in MLNs is full MAP inference. In this setting, given a set of ground literals (the evidence) the goal is to compute the most probable configuration of all unobserved variables (the queries). Two standard approaches for performing MAP inference in MLNs are to employ a strategy based on MaxWalkSAT [22] or to use a cutting plane based strategy [23, 18]. Given a set of ground formulas  $E$ , we write  $\max(\mathcal{M}, E)$  for the set of most probable worlds of the MLN that satisfy  $E$ . For each  $\omega \in \max(\mathcal{M}, E)$ ,  $\sum_{(F, w_F) \in \mathcal{M}} w_F n_F(\omega)$  evaluates to the same value, which we will refer to as  $\text{sat}(\mathcal{M}, E)$ . We define the penalty  $\text{pen}(\mathcal{M}, E)$  of  $E$  as follows:

$$\text{pen}(\mathcal{M}, E) = \text{sat}(\mathcal{M}, \emptyset) - \text{sat}(\mathcal{M}, E)$$

We will sometimes identify possible worlds with the set of literals they make true, writing  $\text{pen}(\mathcal{M}, \omega)$ . We will also write  $\text{pen}(\mathcal{M}, \alpha)$ , with  $\alpha$  a ground formula, as a shorthand for  $\text{pen}(\mathcal{M}, \{\alpha\})$ . We will consider the following inference relation, which has been considered among others in [13]:

$$(\mathcal{M}, E) \vdash_{\text{MAP}} \alpha \quad \text{iff} \quad \forall \omega \in \max(\mathcal{M}, E) : \omega \models \alpha \quad (2)$$

with  $\mathcal{M}$  an MLN,  $\alpha$  a ground formula and  $E$  a set of ground formulas. It can be shown that checking  $(\mathcal{M}, E) \vdash_{\text{MAP}} \alpha$  for a ground network  $\mathcal{M}$  is  $\Delta_2^P$ -complete<sup>2</sup> [4].

### 2.2 POSSIBILISTIC LOGIC

A possibility distribution in a universe  $\Omega$  is a mapping  $\pi$  from  $\Omega$  to  $[0, 1]$ , encoding our knowledge about the possible values that a given variable  $X$  can take; throughout this paper, we will assume that all universes are finite. For each  $x \in \Omega$ ,  $\pi(x)$  is called the possibility degree of  $x$ . By convention, in a state of complete ignorance, we have  $\pi(x) = 1$  for all  $x \in \Omega$ ; conversely, if  $X = x_0$  is known, we have  $\pi(x_0) = 1$  and  $\pi(x) = 0$  for  $x \neq x_0$ . Possibility theory [27, 12] is based on the possibility measure  $\Pi$  and dual necessity measure  $N$ , induced by a possibility distribution  $\pi$  as follows ( $A \subseteq \Omega$ ):

$$\Pi(A) = \max_{a \in A} \pi(a)$$

$$N(A) = 1 - \Pi(\Omega \setminus A)$$

<sup>2</sup>The complexity class  $\Delta_2^P$  contains those decision problems that can be solved in polynomial time on a deterministic Turing machine with access to an NP oracle.

<sup>1</sup><http://arxiv.org/abs/1506.01432>

Intuitively,  $\Pi(A)$  is the degree to which available evidence is compatible with the view that  $X$  belongs to  $A$ , whereas  $N(A)$  is the degree to which available evidence implies that  $X$  belongs to  $A$ , i.e. the degree to which it is certain that  $X$  belongs to  $A$ .

A theory in possibilistic logic [9] is a set of formulas of the form  $(\alpha, \lambda)$ , where  $\alpha$  is a propositional formula and  $\lambda \in [0, 1]$  is a certainty weight. A possibility distribution  $\pi$  satisfies  $(\alpha, \lambda)$  iff  $N(\llbracket \alpha \rrbracket) \geq \lambda$ , with  $N$  the necessity measure induced by  $\pi$  and  $\llbracket \alpha \rrbracket$  the set of propositional models of  $\alpha$ . We say that a possibilistic logic theory  $\Theta$  entails  $(\alpha, \lambda)$ , written  $\Theta \models (\alpha, \lambda)$ , if every possibility distribution which satisfies all the formulas in  $\Theta$  also satisfies  $(\alpha, \lambda)$ . A possibility distribution  $\pi_1$  is called less specific than a possibility distribution  $\pi_2$  if  $\pi_1(\omega) \geq \pi_2(\omega)$  for every  $\omega$ . It can be shown that the set of models of  $\Theta$  always has a least element w.r.t. the minimal specificity ordering, which is called the least specific model  $\pi^*$  of  $\Theta$ . It is easy to see that  $\Theta \models (\alpha, \lambda)$  iff  $\pi^*$  satisfies  $(\alpha, \lambda)$ .

Even though the semantics of possibilistic logic is defined at the propositional level, we will also use first-order formulas such as  $(p(X) \rightarrow q(X, Y), \lambda)$  throughout the paper. As in Markov logic, we will interpret these formulas as abbreviations for a set of propositional formulas, obtained using grounding in the usual way. In particular, we will always assume that first-order formulas are defined w.r.t. a finite set of constants.

The  $\lambda$ -cut  $\Theta_\lambda$  of a possibilistic logic theory  $\Theta$  is defined as follows:

$$\Theta_\lambda = \{\alpha \mid (\alpha, \mu) \in \Theta, \mu \geq \lambda\}$$

It can be shown that  $\Theta \models (\alpha, \lambda)$  iff  $\Theta_\lambda \models \alpha$ , which means that inference in possibilistic logic can straightforwardly be implemented using a SAT solver.

An inconsistency-tolerant inference relation  $\vdash_{poss}$  for possibilistic logic can be defined as follows:

$$\Theta \vdash_{poss} \alpha \quad \text{iff} \quad \Theta_{con(\Theta)} \models \alpha$$

where the consistency level  $con(\Theta)$  of  $\Theta$  is the lowest certainty level  $\lambda$  for which  $\Theta_\lambda$  is satisfiable (among the certainty levels that occur in  $\Theta$ ). Note that all formulas with a certainty level below  $con(\Theta)$  are ignored, even if they are unrelated to any inconsistency in  $\Theta$ . This observation is known as the drowning effect.

We will write  $(\Theta, E) \vdash_{poss} \alpha$ , with  $E$  a set of propositional formulas, as an abbreviation for  $\Theta \cup \{(e, 1) \mid e \in E\} \vdash_{poss} \alpha$ . Despite its conceptual simplicity,  $\vdash_{poss}$  has many desirable properties. Among others, it is closely related to AGM belief revision [8] and default reasoning [2]. It can be shown that checking  $\Theta \vdash_{poss} (\alpha, \lambda)$  is a  $\Theta_2^P$  complete problem<sup>3</sup> [17]. In this paper,  $\vdash_{poss}$  will allow us to capture the non-monotonicity of MAP inference.

<sup>3</sup>The complexity class  $\Theta_2^P$  contains those decision problems

**Example 2.** Let  $\Theta$  consist of the following formulas:

$$\begin{aligned} &(penguin(X) \rightarrow bird(X), 1) \\ &(penguin(X) \rightarrow \neg flies(X), 1) \\ &(bird(X) \rightarrow flies(X), 0.5) \end{aligned}$$

Then we find:

$$\begin{aligned} &(\Theta, \{bird(tweety)\}) \vdash_{poss} flies(tweety) \\ &(\Theta, \{bird(tweety), penguin(tweety)\}) \vdash_{poss} \neg flies(tweety) \end{aligned}$$

In general,  $\vdash_{poss}$  allows us to model rules with exceptions, by ensuring that rules about specific contexts have a higher certainty weight than rules about general contexts.

### 3 ENCODING GROUND NETWORKS

Throughout this section, we will assume that  $\mathcal{M}$  is a ground MLN in which all the weights are strictly positive. This can always be guaranteed for ground MLNs by replacing formulas  $(\alpha, \lambda)$  with  $\lambda < 0$  by  $(\neg\alpha, -\lambda)$ , and by discarding any formula whose weight is 0. For a subset  $X \subseteq \mathcal{M}$ , we write  $X^*$  for the set of corresponding classical formulas, e.g. for  $X = \{(F_1, w_1), \dots, (F_n, w_n)\}$  we have  $X^* = \{F_1, \dots, F_n\}$ . In particular,  $\mathcal{M}^*$  are the classical formulas appearing in the MLN  $\mathcal{M}$ .

The following transformation constructs a possibilistic logic theory that is in some sense equivalent to a given MLN. It is inspired by the probability-possibility transformation from [10].

**Transformation 1.** We define the possibilistic logic theory  $\Theta_{\mathcal{M}}$  corresponding to an MLN  $\mathcal{M}$  as follows:

$$\{(\bigvee X^*, \phi(\neg \bigvee X^*)) \mid X \subseteq \mathcal{M}, \phi(\neg \bigvee X^*) > 0\} \quad (3)$$

where for a propositional formula  $\alpha$ :

$$\phi(\alpha) = \begin{cases} \frac{K + pen(\mathcal{M}, \alpha)}{L} & \text{if } \alpha \text{ satisfies the hard constraints} \\ 1 & \text{otherwise} \end{cases}$$

and the constants  $K$  and  $L$  are chosen such that  $0 = \phi(\top) \leq \phi(\alpha) < 1$  for every  $\alpha$  that satisfies the hard constraints (i.e. the formulas with weight  $+\infty$ ).

In the following we will use the notations  $\phi(\omega)$  for a possible world  $\omega$  and  $\phi(E)$  for a set of formulas  $E$ , defined entirely analogously. Throughout the paper we will also write  $(-K < x < L - K)$ :

$$\lambda_x = \frac{K + x}{L}$$

The correctness of Transformation 1 follows from the next proposition, which is easy to show.

that can be solved in polynomial time on a deterministic Turing machine, by making at most a logarithmic number of calls to an NP oracle.

**Proposition 1.** Let  $\mathcal{M}$  be a ground MLN and  $\Theta_{\mathcal{M}}$  the corresponding possibilistic logic theory. Let  $\pi$  be the least specific model of  $\Theta_{\mathcal{M}}$ . It holds that:

$$\pi(\omega) = 1 - \phi(\omega)$$

**Corollary 1.** Let  $\mathcal{M}$  be a ground MLN and  $\Theta_{\mathcal{M}}$  the corresponding possibilistic logic theory. It holds that for  $\lambda < 1$ :

$$\Theta_{\mathcal{M}} \models (\alpha, \lambda) \quad \text{iff} \quad \text{pen}(\mathcal{M}, \neg\alpha) \geq \lambda L - K$$

and

$$\Theta_{\mathcal{M}} \models (\alpha, 1) \quad \text{iff} \quad \text{pen}(\mathcal{M}, \neg\alpha) = +\infty$$

**Corollary 2.** Let  $\mathcal{M}$  be a ground MLN and  $\Theta_{\mathcal{M}}$  the corresponding possibilistic logic theory. For  $p_{\mathcal{M}}$  the probability distribution induced by  $\mathcal{M}$  and  $\pi$  the least specific model of  $\Theta_{\mathcal{M}}$ , it holds that

$$p_{\mathcal{M}}(\omega_1) > p_{\mathcal{M}}(\omega_2) \quad \text{iff} \quad \pi(\omega_1) > \pi(\omega_2)$$

for all possible worlds  $\omega_1$  and  $\omega_2$ . In particular, it follows that for every propositional formula  $\alpha$  and every set of propositional formulas  $E$ :

$$(\mathcal{M}, E) \vdash_{\text{MAP}} \alpha \quad \text{iff} \quad (\Theta, E) \vdash_{\text{poss}} \alpha \quad (4)$$

**Example 3.** Consider the MLN  $\mathcal{M}$  containing the following formulas:

$$5 : a \rightarrow x \quad 5 : a \rightarrow y \quad 10 : a \wedge b \rightarrow \neg y$$

Then  $\Theta_{\mathcal{M}}$  contains the following formulas:

$$\begin{aligned} \lambda_5 : a \rightarrow x & & \lambda_5 : a \rightarrow y \\ \lambda_{10} : a \wedge b \rightarrow \neg y & & \lambda_{10} : a \rightarrow x \vee y \\ \lambda_{15} : a \wedge b \rightarrow x \vee \neg y & & \end{aligned}$$

It can be verified that:

$$(\Theta_{\mathcal{M}}, \{a\}) \vdash_{\text{poss}} x \wedge y \quad (\Theta_{\mathcal{M}}, \{a, b\}) \vdash_{\text{poss}} x \wedge \neg y$$

An important drawback of the transformation to possibilistic logic is that the number of formulas in  $\Theta_{\mathcal{M}}$  is exponential in  $|\mathcal{M}|$ . This makes the transformation inefficient, and moreover limits the interpretability of the possibilistic logic theory. In general, the exponential size of  $\Theta_{\mathcal{M}}$  cannot be avoided if we want (4) to hold for any  $E$  and  $\alpha$ . However, more compact theories can be found if we focus on specific types of evidence. Sections 3.1 and 3.2 introduce two practical methods to accomplish this.

### 3.1 SELECTIVELY AVOIDING DROWNING

In many applications, we are only interested in particular types of evidence sets  $E$ . For example, we may only

be interested in evidence sets that contain at most  $k$  literals, or in evidence sets that only contain positive literals. In such cases, we can often derive a more compact possibilistic logic theory  $\Theta^{\mathcal{E}}$  as follows. Let  $\mathcal{E}$  be the set of evidence sets that we wish to consider, where each  $E \in \mathcal{E}$  is a set of ground formulas. Given  $E \in \mathcal{E}$  we write  $\mathcal{S}_E$  for the set of all minimal subsets  $\{F_1, \dots, F_l\}$  of  $\mathcal{M}_E^* = \{F \mid F \in \mathcal{M}^*, \text{pen}(\mathcal{M}, \neg F) < \text{pen}(\mathcal{M}, E)\}$  s.t.

$$\text{pen}(\mathcal{M}, \bigwedge E \wedge \neg F_1 \wedge \dots \wedge \neg F_l) > \text{pen}(\mathcal{M}, E) \quad (5)$$

The following transformation constructs a possibilistic logic theory that correctly captures MAP inference for evidence sets in  $\mathcal{E}$ . The basic intuition is that we want to weaken the formulas in  $\mathcal{M}^*$  just enough to ensure that the resulting certainty level prevents them from drowning when the evidence  $E$  becomes available.

**Transformation 2.** Given a ground MLN  $\mathcal{M}$  and a set of evidence sets  $\mathcal{E}$ , we define the possibilistic logic theory  $\Theta_{\mathcal{M}}^{\mathcal{E}}$  as follows:

$$\{(F_1, \phi(\neg F_1)) \mid F_1 \in \mathcal{M}^*\} \quad (6)$$

$$\cup \{(\neg \bigwedge E \vee \bigvee Z, \phi(\bigwedge E \wedge \neg \bigwedge Z)) \mid Z \in \mathcal{S}_E, \quad (7)$$

$$E \in \mathcal{E}\} \cup \{(\neg \bigwedge E, \phi(\bigwedge E)) \mid E \in \mathcal{E}\} \quad (8)$$

If  $\mathcal{M}$  is clear from the context, we will omit the subscript in  $\Theta_{\mathcal{M}}^{\mathcal{E}}$ . The formulas in (6) are the direct counterpart of the MLN. Intuitively, there are two reasons why these formulas are not sufficient. First, due to the drowning effect, formulas  $F$  such that  $\text{pen}(\mathcal{M}, \neg F) < \text{pen}(\mathcal{M}, E)$  will be ignored under the evidence  $E$ . In such cases we should look at minimal ways to weaken these formulas such that the certainty level of the resulting formula is sufficient to avoid drowning under the evidence  $E$ . This is accomplished by adding the formulas in (7). Second, as  $\Theta^{\mathcal{E}}$  contains less information than  $\Theta_{\mathcal{M}}$ , we need to ensure that the consistency level for  $\Theta^{\mathcal{E}}$  is never lower than the consistency level for  $\Theta_{\mathcal{M}}$ , given an evidence set  $E \in \mathcal{E}$ . To this end,  $\Theta^{\mathcal{E}}$  includes the formulas in (8). The following example illustrates why these formulas are needed.

**Example 4.** Consider the following MLN  $\mathcal{M}$ :

$$\begin{aligned} 3 : u & & 2 : a & & 10 : (a \vee b) \wedge (u \vee v) \rightarrow \neg x \\ 2 : b & & 1 : v & & \end{aligned}$$

and let  $\mathcal{E} = \{\{x\}\}$ , i.e. the only evidence set in which we are interested is  $\{x\}$ . It holds that

$$\mathcal{S}_E = \{\{a, u\}, \{b, u\}, \{a, v\}, \{b, v\}\} \quad (9)$$

and  $\Theta^{\mathcal{E}} = \Theta \cup \Psi \cup \Gamma$ , where:

$$\Theta = \{(u, \lambda_3), (a, \lambda_2), ((a \vee b) \wedge (u \vee v) \rightarrow \neg x, \lambda_{10}), (b, \lambda_2), (v, \lambda_1)\}$$

$$\Psi = \{(a \vee u \vee \neg x, \lambda_6), (b \vee u \vee \neg x, \lambda_6), (a \vee v \vee \neg x, \lambda_5), (b \vee v \vee \neg x, \lambda_5)\}$$

$$\Gamma = \{(\neg x, \lambda_4)\}$$

It is easy to verify that  $(\Theta \cup \Psi, \{x\}) \vdash_{poss} u$  whereas  $(\mathcal{M}, \{x\}) \not\vdash_{MAP} u$  and  $(\Theta \cup \Psi \cup \Gamma, \{x\}) \not\vdash_{poss} u$ .

We now prove the correctness of Transformation 2.

**Proposition 2.** *For any formula  $\alpha$  and any evidence set  $E \in \mathcal{E}$ , it holds that  $(\Theta_{\mathcal{M}}, E) \vdash_{poss} \alpha$  iff  $(\Theta^{\mathcal{E}}, E) \vdash_{poss} \alpha$ .*

*Proof.* Let us introduce the following notation:

$$\begin{aligned}\lambda_E &= \text{con}(\Theta_{\mathcal{M}} \cup \{(e, 1) \mid e \in E\}) \\ \lambda_E^{\mathcal{E}} &= \text{con}(\Theta^{\mathcal{E}} \cup \{(e, 1) \mid e \in E\}) \\ A &= (\Theta_{\mathcal{M}} \cup \{(e, 1) \mid e \in E\})_{\lambda_E} \\ A^E &= (\Theta^{\mathcal{E}} \cup \{(e, 1) \mid e \in E\})_{\lambda_E^{\mathcal{E}}}\end{aligned}$$

We need to show that  $A$  is equivalent to  $A^E$ , for any  $E \in \mathcal{E}$ .

By Corollary 1, we know that every formula  $(\alpha, \lambda)$  in  $\Theta_{\mathcal{M}}^{\mathcal{E}}$  is entailed by  $\Theta_{\mathcal{M}}$ , hence  $\lambda_E^{\mathcal{E}} \leq \lambda_E$ . Since  $\lambda_E$  is the smallest certainty level from  $\Theta_{\mathcal{M}}$  which is strictly higher than  $\phi(E)$ , it follows that  $A^E$  contains every formula which appears in  $\Theta^{\mathcal{E}}$  with a weight that is strictly higher than  $\phi(E)$ . Moreover, since  $\Theta^{\mathcal{E}}$  by construction contains  $(\neg \wedge E, \phi(\wedge E))$ , we find that  $A^E$  can only contain such formulas:

$$A^E = E \cup \{\alpha \mid (\alpha, \lambda) \in \Theta^{\mathcal{E}}, \lambda > \phi(E)\} \quad (10)$$

It follows that  $A \models A^E$ .

Let  $G_1 \vee \dots \vee G_s$  be a formula from  $A$ . From Corollary 1 we know that:

$$\text{pen}(\mathcal{M}, \neg G_1 \wedge \dots \wedge \neg G_s) > \text{pen}(\mathcal{M}, E)$$

and a fortiori

$$\text{pen}(\mathcal{M}, E \wedge \neg G_1 \wedge \dots \wedge \neg G_s) > \text{pen}(\mathcal{M}, E)$$

This means that for any formula  $G_1 \vee \dots \vee G_s$  in  $A$ , either  $\text{pen}(\mathcal{M}, \neg G_i) > \text{pen}(\mathcal{M}, E)$  for some  $i$  or  $\mathcal{S}_E$  contains a subset  $\{H_1, \dots, H_r\}$  of  $\{G_1, \dots, G_s\}$ . Then  $\Theta^{\mathcal{E}}$  contains either  $G_i$  or the formula  $\neg E \vee H_1 \dots \vee H_r$  with a weight which is strictly higher than  $\phi(E)$  and thus either  $G_i$  or  $\neg E \vee H_1 \dots \vee H_r$  belongs to  $A^E$ . In both cases we find  $A^E \models G_1 \vee \dots \vee G_s$ . We conclude  $A^E \models A$ .  $\square$

An alternative, which would make the approach in this section closer to the standard encoding in (1), is to define  $\mathcal{S}'_E$  as the set of minimal subsets  $\{F_1, \dots, F_l\}$  of  $\mathcal{M}_E^*$  such that

$$\text{pen}(\mathcal{M}, \neg F_1 \wedge \dots \wedge \neg F_l) > \text{pen}(\mathcal{M}, E) \quad (11)$$

and then replace the formulas in (7) by

$$\{(\bigvee Z, \phi(\neg \bigwedge Z)) \mid Z \in \mathcal{S}'_E\} \quad (12)$$

The advantage of (7), however, is that we can expect many of the sets in  $\mathcal{S}_E$  to be singletons. To see why this is

the case, first note that for each world  $\omega$  in  $\max(\mathcal{M}, E)$ , the set of formulas  $\mathcal{Y} \subseteq \mathcal{M}^*$  satisfied by  $\omega$  is such that  $\text{pen}(\mathcal{M}, \neg \bigvee (\mathcal{M}^* \setminus \mathcal{Y}))$  is minimal among all sets  $\mathcal{Y}' \subseteq \mathcal{M}^*$  for which  $E \wedge \bigwedge \mathcal{Y}'$  is consistent. Let us write  $\text{Cons}_E(\mathcal{M})$  for the set of all these maximally consistent subsets of  $\mathcal{M}^*$ . Note that  $\max(\mathcal{M}, E) = \llbracket \bigvee \{ \bigwedge \mathcal{Y} \mid \mathcal{Y} \in \text{Cons}_E(\mathcal{M}) \} \rrbracket$ .

**Lemma 1.** *For a set of formulas  $\{F_1, \dots, F_l\} \subseteq \mathcal{M}^*$  it holds that  $\text{pen}(\mathcal{M}, E \wedge \neg F_1 \wedge \dots \wedge \neg F_l) > \text{pen}(\mathcal{M}, E)$  iff  $\{F_1, \dots, F_l\} \cap \mathcal{Y} \neq \emptyset$  for every  $\mathcal{Y}$  in  $\text{Cons}_E(\mathcal{M})$ .*

**Corollary 3.** *Let  $\text{Cons}_E(\mathcal{M}) = \{\mathcal{Y}_1, \dots, \mathcal{Y}_s\}$ . It holds that  $\mathcal{S}_E$  consists of the subset-minimal elements of  $\{\{y_1, \dots, y_s\} \mid y_1 \in \mathcal{Y}_1 \cap \mathcal{M}_E^*, \dots, y_s \in \mathcal{Y}_s \cap \mathcal{M}_E^*\}$ .*

**Example 5.** *Consider again the MLN  $\mathcal{M}$  from Example 4 and let  $E = \{x\}$ . It holds that  $\text{Cons}_E(\mathcal{M}) = \{\mathcal{Y}_1, \mathcal{Y}_2\}$ , where*

$$\begin{aligned}\mathcal{Y}_1 &= \{(a \vee b) \wedge (u \vee v) \rightarrow \neg x, a, b\} \\ \mathcal{Y}_2 &= \{(a \vee b) \wedge (u \vee v) \rightarrow \neg x, u, v\} \\ \mathcal{M}_E^* &= \{a, b, u, v\}\end{aligned}$$

From Corollary 3, it follows that  $\mathcal{S}_E$  is given by (9).

In practice,  $\text{Cons}_E(\mathcal{M})$  will often contain a single element, in which case all the elements of  $\mathcal{S}_E$  will be singletons.

### 3.2 MAP INFERENCE AS DEFAULT REASONING

A large number of approaches has been proposed for reasoning with a set of default rules of the form “if  $\alpha$  then typically  $\beta$ ” [15, 19, 14]. At the core, each of the proposed semantics corresponds to the intuition that a set of default rules imposes a preference order on possible worlds, where “if  $\alpha$  then  $\beta$ ” means that  $\beta$  is true in the most preferred models of  $\alpha$ . The approaches from [15] and [19] can be elegantly captured in possibilistic logic [2], by interpreting the default rule as the constraint  $\Pi(\alpha \wedge \beta) > \Pi(\alpha \wedge \neg \beta)$ . In Markov logic, the same constraint on the ordering of possible worlds can be expressed by imposing the constraint  $(\mathcal{M}, \alpha) \vdash_{MAP} \beta$ . In other words, we can view the MAP consequences of an MLN as a set of default rules, and encode these default rules in possibilistic logic. The following transformation is based on this idea.

**Transformation 3.** *Given a ground MLN  $\mathcal{M}$  and a positive integer  $k$ , we construct a possibilistic logic theory  $\Theta_{\mathcal{M}}^k$  as follows:*

- For each hard rule  $F$  from  $\mathcal{M}$ , add  $(F, 1)$  to  $\Theta_{\mathcal{M}}^k$ .
- For each set of literals  $E$  such that  $0 \leq |E| \leq k$ , let  $X = \{x \mid (\mathcal{M}, E) \vdash_{MAP} x\}$  be the set of literals that are true in all the most plausible models of  $E$ . Unless there is a literal  $y \in E$  such that  $\bigwedge (E \setminus \{y\}) \vdash_{MAP} y$ , add

$$\left( \bigwedge E \rightarrow \bigwedge X, \lambda_E \right)$$

to  $\Theta_{\mathcal{M}}^k$ , where  $\lambda_E = \phi(\bigwedge E)$ . If  $\text{pen}(\mathcal{M}, E) > \text{pen}(\mathcal{M}, \emptyset)$ , add also

$$(\neg(\bigwedge E \wedge \bigwedge X), \lambda'_E) \quad (13)$$

where  $\lambda'_E$  is the certainty level just below  $\lambda_E$  in  $\Theta_{\mathcal{M}}^k$ , i.e.  $\lambda_{E'} = \max\{\lambda_F \mid \lambda_F < \lambda_E, |F| \leq k\}$ .

If  $\mathcal{M}$  is clear from the context, we will omit the subscript in  $\Theta_{\mathcal{M}}^k$ . The possibilistic encoding of default rules used in Transformation 3 is similar in spirit to the method from [2], which is based on the Z-ranking from [19]. However, because  $p_{\mathcal{M}}$  already provides us with a model of the default rules, we can directly encode default rules in possibilistic logic, without having to rely on the Z-ranking. Also note that although the method is described in terms of an MLN, it can be used for encoding any ranking on possible worlds (assuming a finite set of atoms).

As illustrated in the following example, (13) is needed to avoid deriving too much, serving a similar purpose to (8) in the approach from Section 3.1.

**Example 6.** Consider the following MLN  $\mathcal{M}$ :

$$2 : \neg a \vee b \quad 2 : a \vee b \quad 1 : a \vee \neg b$$

Then  $\Theta^1 = \Theta \cup \Psi$ , where

$$\begin{aligned} \Theta &= \{(\top \rightarrow a \wedge b, \lambda_0), (\neg a \rightarrow b, \lambda_1), (\neg b \rightarrow \top, \lambda_2)\} \\ \Psi &= \{(b, \lambda_1), (a \vee \neg b, \lambda_0)\} \end{aligned}$$

We find  $(\Theta, \{\neg b\}) \vdash_{\text{poss}} a$  while  $(\mathcal{M}, \{\neg b\}) \not\vdash_{\text{MAP}} a$ . Accordingly, we have  $(\Theta \cup \Psi, \{\neg b\}) \not\vdash_{\text{poss}} a$ .

Transformations 2 and 3 have complementary strengths. For example, Transformation 2 may lead to more compact theories for relatively simple MLNs, e.g. if for most of the considered evidence sets, there is a unique set of formulas from the MLN that characterizes the most probable models of the evidence (cf. Lemma 1). On the other hand, Transformation 3 may lead to substantially more compact theories in cases where the number of formulas is large relative to the number of atoms.

We now show the correctness of Transformation 3.

**Proposition 3.** Let  $\mathcal{M}$  be an MLN,  $k$  a positive integer and  $\Theta^k$  the proposed possibilistic logic encoding of  $\mathcal{M}$ . Furthermore, let  $E$  and  $C$  be sets of literals such that  $|E| + |C| \leq k + 1$ . It holds that  $(\mathcal{M}, E) \vdash_{\text{MAP}} \bigvee C$  if and only if  $(\Theta^k, E) \vdash_{\text{poss}} \bigvee C$ .

Before we prove Proposition 3, we present a number of lemmas. In the lemmas and proofs below,  $\mathcal{M}$  will always be an MLN,  $\Theta^k$  will be the corresponding possibilistic logic theory and  $k$  will be the maximum size of the evidence sets considered in the translation.

**Lemma 2.** If  $E$  is a set of literals,  $|E| \leq k$ ,  $\lambda = \phi(E)$  and  $(\mathcal{M}, E) \vdash_{\text{MAP}} x$  then

$$\left\{ \left( \bigwedge E' \rightarrow \bigwedge X \right) \in \Theta_{\lambda}^k \text{ s.t. } |E'| \leq |E| \right\} \vdash \bigwedge E \rightarrow x$$

**Lemma 3.** If  $\phi(\omega) \leq \lambda$  then  $\omega$  is a model of  $\Theta_{\lambda}^k$ .

*Proof.* If there were a formula  $F = (\bigwedge E) \rightarrow (\bigwedge X)$  in  $\Theta_{\lambda}^k$  that was not satisfied by  $\omega$ , then its body would have to be true in  $\omega$  but then necessarily

$$\lambda \leq \phi(E) \leq \phi(\omega) \leq \lambda.$$

The first inequality follows from the fact that, by the construction of  $\Theta^k$ , if the certainty weight of  $F$  is at least  $\lambda$  then it must be the case that  $\phi(E) \geq \lambda$ . The second inequality follows from the fact that  $\omega$  was assumed to be a model of  $\bigwedge E$ . It follows that:

$$\text{pen}(\mathcal{M}, \omega) = \text{pen}(\mathcal{M}, E).$$

However, this would mean that  $\omega$  is also a most probable world of  $(\mathcal{M}, E)$ , but then  $\omega \models F$  by construction of  $\Theta^k$ .

If there were an unsatisfied formula  $F = \neg(\bigwedge E \wedge \bigwedge X)$  in  $\Theta_{\lambda}^k$  then by construction we would have  $\phi(E \cup X) > \lambda$ . However, from  $\omega \models \bigwedge E \wedge \bigwedge X$  we find  $\phi(E \cup X) \leq \phi(\omega) \leq \lambda$ , a contradiction.

Since all formulas in  $\Theta^k$  are of the two considered types, it follows that all formulas from  $\Theta^k$  whose certainty weight is at least  $\lambda$  must be satisfied in  $\omega$ .  $\square$

**Lemma 4.** If  $(\mathcal{M}, E) \vdash_{\text{MAP}} (y_1 \vee \dots \vee y_m)$  then

- (i) for any  $i$ , either  $(\mathcal{M}, E \cup \{\neg y_i\}) \vdash_{\text{MAP}} (y_1 \vee \dots \vee y_{i-1} \vee y_{i+1} \vee \dots \vee y_m)$  or  $(\mathcal{M}, E) \vdash_{\text{MAP}} y_i$ ,
- (ii) there exist a  $j$  and a set  $\{y'_1, \dots, y'_m\} \subseteq \{y_1, \dots, y_m\} \setminus \{y_j\}$  such that  $(\mathcal{M}, E \cup \{\neg y'_1, \dots, \neg y'_m\}) \vdash_{\text{MAP}} y_j$ .

**Lemma 5.** If  $|C| + |E| \leq k + 1$ , and  $\lambda = \phi(E)$  then  $\Theta_{\lambda}^k \cup E \vdash \bigvee C$  if and only if  $(\mathcal{M}, E) \vdash_{\text{MAP}} \bigvee C$ .

We now turn to the proof of Proposition 3.

*Proof of Proposition 3.* Let  $E$  be an evidence set such that  $|E| \leq k$  and let  $\lambda = \phi(E)$ . Given Lemma 5, it is sufficient to show that  $\text{con}(\Theta^k, E) = \lambda$ . It follows from Lemma 3 that  $\text{con}(\Theta^k, E) \leq \lambda$ . Let  $X = \{x \mid (\mathcal{M}, E) \vdash_{\text{MAP}} x\}$  be the set of literals which can be derived from  $(\mathcal{M}, E)$  using MAP inference. By construction,  $\Theta^k$  contains a formula  $\neg(\bigwedge E \wedge \bigwedge X)$  with a certainty weight which is just below  $\lambda$ . Specifically, for  $\lambda' < \lambda$  we either have  $\Theta_{\lambda'}^k = \Theta_{\lambda}^k$ , or  $\Theta_{\lambda'}^k \models \neg \bigwedge E$ , from which we find  $\text{con}(\Theta^k, E) = \lambda$ .  $\square$

It is of interest to remove any formulas in  $\Theta^k$  that are redundant, among others because this is likely to make the theory easier to interpret. Although we can use possibilistic logic inference to identify redundant formulas, in some cases we can avoid adding the redundant formulas altogether. For example, in the transformation procedure, we do not add any rules for  $E$  if it holds that  $E \setminus \{y\} \vdash_{MAP} y$  for some  $y \in E$ . This pruning rule is the counterpart of the cautious monotonicity property, which is well-known in the context of default reasoning [15]. Any ranking on possible worlds also satisfies the stronger rational monotonicity property, which translated to our setting states that when  $(\mathcal{M}, E \setminus \{y\}) \vdash_{MAP} x$  and  $(\mathcal{M}, E \setminus \{y\}) \not\vdash_{MAP} \neg y$  it holds that  $(\mathcal{M}, E) \vdash_{MAP} x$ . Accordingly, when processing the evidence set  $E$  in the transformation procedure, instead of  $(\bigwedge E \rightarrow \bigwedge X, \lambda_E)$  it is sufficient to add the following rule:

$$\left( \bigwedge E \rightarrow \bigwedge (X \setminus X_0), \lambda_E \right)$$

where

$$X_0 = \{x \mid E \setminus \{y\} \vdash_{MAP} x \text{ and } E \setminus \{y\} \not\vdash_{MAP} \neg y\}$$

The correctness of this pruning step follows from the following proposition.

**Proposition 4.** *Let  $x$  and  $y$  be literals. If  $|E| < k$ ,  $(\mathcal{M}, E) \vdash_{MAP} x$  and  $(\mathcal{M}, E) \not\vdash_{MAP} \neg y$  then:*

$$\Theta^k \setminus \{F\} \models \left( \bigwedge E \wedge y \rightarrow x, \lambda_{E \cup \{y\}} \right)$$

where  $F$  is the formula in  $\Theta^k$  corresponding to the evidence set  $E \cup \{y\}$ , i.e.:

$$F = \bigwedge (E \cup \{y\}) \rightarrow \bigwedge \{x \mid (\mathcal{M}, E \cup \{y\}) \vdash_{MAP} x\}$$

*Proof.* If  $(\mathcal{M}, E) \vdash_{MAP} x$  and  $(\mathcal{M}, E) \not\vdash_{MAP} \neg y$  then  $(\mathcal{M}, E \cup \{y\}) \vdash_{MAP} x$  and  $pen(\mathcal{M}, E) = pen(\mathcal{M}, E \cup \{y\}) = pen(\mathcal{M}, E \cup \{x, y\})$ . Therefore using Lemma 2, we find that  $\Theta^k_{\lambda_{E \cup \{y\}}} \vdash \bigwedge E \rightarrow x$ . From Lemma 2, it furthermore follows that  $\bigwedge E \rightarrow x$  can be derived from rules with antecedents of length at most  $|E|$ . In particular, we find that  $\bigwedge E \rightarrow x$  can be derived without using the formula  $\bigwedge E \wedge y \rightarrow \bigwedge X$ .  $\square$

Finally, note that formulas of the form (13) can be omitted when  $\lambda'_E = \lambda_{(E \setminus \{y\})}$  for some  $y \in E$ . Indeed, in such a case we find from  $pen(\mathcal{M}, E \setminus \{y\}) < pen(\mathcal{M}, E)$  that  $(\mathcal{M}, E \setminus \{y\}) \vdash_{MAP} \neg y$ , hence (13) will be entailed by a formula of the form  $(\bigwedge (E \setminus \{y\}) \rightarrow \bigwedge X, \lambda_{E \setminus \{y\}})$  in  $\Theta^k$ .

## 4 ENCODING NON-GROUND NETWORKS

We now provide the counterpart to the construction from Section 3.2 for non-ground MLNs. The first-order nature

of MLNs often leads to distributions with many symmetries which can be exploited by lifted inference methods [20]. We can similarly exploit these symmetries for constructing more compact possibilistic logic theories from MLNs.

For convenience, in the possibilistic logic theories, we will use *typed* formulas. For instance, when we have the formula  $\alpha = owns(person : X, thing : Y)$  and the set of constants of the type *person* is  $\{alice, bob\}$  and the set of constants of the type *thing* is  $\{car\}$  then  $\alpha$  corresponds to the ground formulas  $owns(alice, car)$  and  $owns(bob, car)$ . In cases where there is only one type, we will not write it explicitly.

Two typed formulas  $F_1$  and  $F_2$  are said to be isomorphic when there is a type-respecting substitution  $\theta$  of the variables of  $F_1$  such that  $F_1\theta \equiv F_2$  (where  $\equiv$  denotes equivalence of logical formulas). Two MLNs  $\mathcal{M}_1$  and  $\mathcal{M}_2$  are said to be isomorphic, denoted by  $\mathcal{M}_1 \approx \mathcal{M}_2$ , if there is a bijection  $i$  from formulas of  $\mathcal{M}_1$  to formulas of  $\mathcal{M}_2$  such that for  $i(F, w) = (F', w')$  it holds that  $w = w'$  and the formulas  $F$  and  $F'$  are isomorphic. When  $j$  is a permutation of a subset of constants from  $\mathcal{M}$  then  $j(\mathcal{M})$  denotes the MLN obtained by replacing any constant  $c$  from the subset by its image  $j(c)$ .

Given a non-ground MLN  $\mathcal{M}$ , we can first identify sets of constants which are *interchangeable*, where a set of constants  $\mathcal{C}_t$  is said to be interchangeable if  $j(\mathcal{M}) \approx \mathcal{M}$  for any permutation  $j$  of the constants in  $\mathcal{C}_t$ . Note that to check whether a set of constants  $\mathcal{C}_t$  is interchangeable, it is sufficient to check that  $j(\mathcal{M}) \approx \mathcal{M}$  for those permutations which swap just two constants from  $\mathcal{C}_t$ . For every maximal set  $\mathcal{C}_t$  of interchangeable constants, we introduce a new type  $t$ . For a constant  $c$ , we write  $\tau(c)$  to denote its type. When  $F$  is a ground formula, *variabilize*( $F$ ) denotes the following formula:

$$\bigwedge \{V_c \neq V_d \mid c, d \in const(F), \tau(c) = \tau(d)\} \rightarrow F'$$

where  $const(F)$  is the set of constants appearing in  $F$  and  $F'$  is obtained from  $F$  by replacing all constants  $c$  by a new variable  $V_c$  of type  $\tau(c)$ .

**Transformation 4.** *Given an MLN  $\mathcal{M}$  and a positive integer  $k$ , we construct a possibilistic logic theory  $\Theta^k_{\mathcal{M}}$  as follows:*

- For each hard rule  $F$  from  $\mathcal{M}$ , add  $(F, 1)$  to  $\Theta^k_{\mathcal{M}}$ .
- For each set of literals  $E$  such that  $0 \leq |E| \leq k$ , let  $X = \{x \mid (\mathcal{M}, E) \vdash_{MAP} x\}$ . For all  $x \in X$ , unless there is a literal  $y \in E$  such that  $(\mathcal{M}, E \setminus \{y\}) \vdash_{MAP} y$  and unless  $\Theta^k_{\mathcal{M}}$  already contains a formula isomorphic to *variabilize*( $\bigwedge E \rightarrow x$ ), add

$$\left( \text{variabilize} \left( \bigwedge E \rightarrow x \right), \lambda_E \right)$$

to  $\Theta_{\mathcal{M}}^k$ . If  $\text{pen}(\mathcal{M}, E) > \text{pen}(\mathcal{M}, \emptyset)$  and  $\Theta_{\mathcal{M}}^k$  does not already contain a formula isomorphic to  $\text{variabilize}(\neg(\bigwedge E \wedge \bigwedge X))$ , add also

$$\left(\text{variabilize}\left(\neg\left(\bigwedge E \wedge \bigwedge X\right)\right), \lambda'_E\right) \quad (14)$$

where  $\lambda'_E$  is the certainty level just below  $\lambda_E$  in  $\Theta_{\mathcal{M}}^k$ .

As before, we will usually omit the subscript in  $\Theta_{\mathcal{M}}^k$ . We can show that after grounding,  $\Theta^k$  is equivalent to the theory that would be obtained by first grounding the MLN and then applying the method from Section 3.2. The correctness proof is provided in the online appendix.

Our implementation<sup>4</sup> of Transformation 4 relies on an efficient implementation of inference in possibilistic logic and Markov logic, efficient generation of non-redundant candidate evidence sets and efficient filtering of isomorphic formulas. For MAP inference in MLNs, we used a cutting-plane inference algorithm based on a SAT-based optimization. For inference in possibilistic logic, we also used cutting-plane inference in order to avoid having to ground the whole theory. To find the ground rules that need to be added by the cutting-plane method, we used a modified querying system from [16]. For solving and optimizing the resulting ground programs, we used the SAT4J library [3].

Note that to check whether  $\Theta_{\lambda}^k \vdash F$ , where  $F$  is a (not necessarily ground) clause, it is sufficient to find one (type-respecting) grounding  $\theta$  of  $F$ , and check whether  $\Theta_{\lambda}^k \cup \{\neg(F\theta)\}$  is inconsistent. In this way, we can check whether a rule is implied by  $\Theta^k$  without grounding the whole theory because, as for MLNs, inference in non-ground possibilistic logic theories can be carried out by cutting-plane inference methods.

We implemented the transformation as a modification of the standard best-first search (BFS) algorithm which constructs incrementally larger candidate evidence sets, checks their MAP consequences and adds the respective rules to the possibilistic logic theory being constructed. Like the standard BFS algorithm it uses a hash-table based data structure *closed*, in which already processed evidence sets are stored. In order to avoid having to check isomorphism with every evidence set in *closed*, each time a new evidence set is considered, the stored evidence sets are enriched by fingerprints which contain some invariants, guaranteeing that no two variabilized evidence sets with different fingerprints are isomorphic. In this way, we can efficiently check for a given evidence set  $E$  whether there is a previously generated evidence set  $E'$  such that  $\text{variabilize}(E)$  and  $\text{variabilize}(E')$  are isomorphic.

As a final remark, we note that for the non-ground transformation, it may be preferable to replace any

<sup>4</sup>The implementation can be downloaded from: <https://github.com/supertweety/mln2poss>.

rule  $(\text{variabilize}(\neg(\bigwedge E \wedge \bigwedge X)), \lambda'_E)$  by the rule  $(\text{variabilize}(\neg\bigwedge E), \lambda'_E)$ . The reason is that the former rules may often become too long in the non-ground case. On the other hand, for the ground transformation, the advantage of the longer rules is that they will often be the same for different sets  $E$ , which, in effect, means a smaller number of rules in the possibilistic logic theory. The correctness of this alternative to Transformation 4 is also shown in the online appendix.

## 5 ILLUSTRATIVE EXAMPLES

The first example is a variation on a classical problem from non-monotonic reasoning. Here, we want to express that birds generally fly, but heavy antarctic birds do not fly, unless they have a jet pack. The MLN which we will convert into possibilistic logic contains the following rules:  $10 : \text{bird}(X) \rightarrow \text{flies}(X)$ ,  $1 : \text{antarctic}(X) \rightarrow \neg\text{flies}(X)$ ,  $10 : \text{heavy}(X) \rightarrow \neg\text{flies}(X)$ ,  $100 : \text{hasJetPack}(X) \rightarrow \text{flies}(X)$ . When presented with this MLN, Transformation 4 produces the following possibilistic logic theory.

$$\begin{aligned} &(\neg\text{antarctic}(X) \vee \neg\text{flies}(X), \lambda_0) \\ &(\neg\text{bird}(X) \vee \text{flies}(X), \lambda_0) \\ &(\neg\text{heavy}(X) \vee \neg\text{flies}(X), \lambda_0) \\ &(\text{flies}(X) \vee \neg\text{hasJetPack}(X), \lambda_0) \\ &(\neg\text{bird}(X) \vee \text{flies}(X) \vee \text{hasJetPack}(X), \lambda_1) \\ &(\neg\text{heavy}(X) \vee \text{antarctic}(X) \vee \neg\text{flies}(X), \lambda_1) \\ &(\neg\text{bird}(X) \vee \neg\text{heavy}(X), \lambda_1) \\ &(\neg\text{antarctic}(X) \vee \neg\text{heavy}(X) \vee \neg\text{flies}(X), \lambda_{10}) \\ &(\text{flies}(X) \vee \neg\text{hasJetPack}(X) \vee \text{bird}(X), \lambda_{11}) \\ &(\neg\text{bird}(X) \vee \text{flies}(X) \vee \neg\text{hasJetPack}(X), \lambda_{100}) \end{aligned}$$

Let us consider the evidence set  $E = \{\text{bird}(\text{tweety}), \text{heavy}(\text{tweety})\}$ . Then the levels  $\lambda_0$  and  $\lambda_1$  drown because of the inconsistency with the rule  $(\neg\text{bird}(X) \vee \neg\text{heavy}(X), \lambda_1)$  which was produced as one of the rules (14). We can see from the rest of the possibilistic logic theory that unless we add either  $\text{antarctic}(\text{tweety})$  or  $\text{hasJetPack}(\text{tweety})$ , we cannot say anything about whether *tweety* flies or not. It can be verified that the same is true also for the respective MLN.

The second example consists of formulas from a classical MLN about smokers. There are three predicates in this MLN: a binary predicate  $f(A, B)$  denoting that  $A$  and  $B$  are friends, and two unary predicates  $s(A)$  and  $c(A)$  denoting that  $A$  smokes and that  $A$  has cancer, respectively. The MLN contains the following hard rules:  $\neg f(A, B) \vee f(B, A)$  and  $\neg f(A, A)$ . In addition, we have two soft rules. The first soft rule  $10 : \neg s(A) \vee \neg f(A, B) \vee s(B)$  states that if  $A$  and  $B$  are friends and  $A$  smokes then  $B$  is more likely to smoke too. The second rule  $10 : \neg s(A) \vee c(A)$  states that smoking increases the likelihood of cancer. The following possi-

bilistic logic theory was obtained using Transformation 4 with  $k = 4$ .

$$\begin{aligned}
& (s(B) \vee \neg f(A, B) \vee \neg s(A) \vee \neg \text{alldiff}(A, B), \lambda_0) \\
& \quad (\neg s(A) \vee c(A), \lambda_0) \\
& (\neg f(C, B) \vee \neg f(A, B) \vee s(A) \vee s(C) \\
& \quad \vee \neg \text{alldiff}(A, B, C) \vee \neg s(B), \lambda_{10}) \\
& (\neg f(C, B) \vee \neg s(A) \vee \neg f(A, C) \vee s(C) \\
& \quad \vee \neg \text{alldiff}(A, B, C) \vee \neg s(B), \lambda_{10}) \\
& (\neg s(A) \vee \neg f(C, A) \vee s(C) \vee c(B) \\
& \quad \vee \neg \text{alldiff}(A, B, C) \vee \neg s(B), \lambda_{10}) \\
& (\neg s(A) \vee c(A) \vee c(B) \vee \neg s(B) \vee \neg \text{alldiff}(A, B), \lambda_{10}) \\
& (s(B) \vee \neg f(A, B) \vee \neg s(A) \vee c(A) \vee \neg \text{alldiff}(A, B), \lambda_{10}) \\
& \quad (\neg f(A, B) \vee f(B, A), 1) \\
& \quad (\neg f(A, A), 1)
\end{aligned}$$

At the lowest level  $\lambda_0$  we find the counterparts of the soft rules from the MLN, whereas at level 1 we find the hard rules. At the intermediate level we intuitively find weakened rules from the MLN. For instance, the rule  $(\neg s(A) \vee c(A) \vee c(B) \vee \neg s(B) \vee \neg \text{alldiff}(A, B), \lambda_1)$  can be interpreted as: if  $A$  and  $B$  smoke then at least one of them has cancer. It is quite natural that this rule has higher certainty weight than the rule: if  $A$  smokes then  $A$  has cancer.

A final, more elaborate example is provided in the online appendix.

## 6 RELATED WORK

One line of related work focuses on extracting a comprehensible model from another learned model that is difficult or impossible to interpret. A seminal work in this area is the TREPAN [5] algorithm. Given a trained neural network and a data set, TREPAN learns a decision tree to mimic the predictions of the neural network. In addition to producing interpretable output, this algorithm was shown to learn accurate models that faithfully mimicked the neural network's predictions. More recent research has focused on approximating complex ensemble classifiers with a single model. For example, Popovic et al. [21] proposed a method for learning a single decision tree that mimics the predictions of a random forest.

While, to the best of our knowledge, this is the first paper that studies the relation between Markov logic and possibilistic logic, the links between possibility theory and probability theory have been widely studied. For example, [10] has proposed a probability-possibility transformation based on the view that a possibility measure corresponds to a particular family of probability measures. Dempster-Shafer evidence theory [25] has also been used to provide a probabilistic interpretation to possibility degrees. In particular, a possibility distribution can be interpreted as the contour

function of a mass assignment; see [11] for details. In [13] it is shown how the probability distribution induced by a penalty logic theory corresponds to the contour function of a mass assignment, which suggests that it is indeed natural to interpret this probability distribution as a possibility distribution. Several other links between possibility theory and probability theory have been discussed in [6].

In this paper, we have mainly focused on MAP inference. An interesting question is whether it would be possible to construct a (possibilistic) logic base that captures the set of accepted beliefs encoded by a probability distribution, where  $A$  is accepted if  $P(A) > P(\neg A)$ . Unfortunately, the results in [7] show that this is only possible for the limited class of so-called big-stepped probability distributions. In practice, this means that we would have to define a partition of the set of possible worlds, such that the probability distribution over the partition classes is big-stepped, and only capture the beliefs that are encoded by the latter, less informative, probability distribution. A similar approach was taken in [1] to learn default rules from data.

## 7 CONCLUSIONS

This paper has focused on how a Markov logic network  $\mathcal{M}$  can be encoded in possibilistic logic. We started from the observation that it is always possible to construct a possibilistic logic theory  $\Theta_{\mathcal{M}}$  that is equivalent to  $\mathcal{M}$ , in the sense that the probability distribution induced by  $\mathcal{M}$  is isomorphic to the possibility distribution induced by  $\Theta_{\mathcal{M}}$ . As a result, applying possibilistic logic inference to  $\Theta_{\mathcal{M}}$  yields the same conclusions as applying MAP inference to  $\mathcal{M}$ . Although the size of  $\Theta_{\mathcal{M}}$  is exponential in the number of formulas in  $\mathcal{M}$ , we have shown how more compact theories can be obtained in cases where we can put restrictions on the types of evidence that need to be considered (e.g. small sets of literals).

Our main motivation has been to use possibilistic logic as a way to make explicit the assumptions encoded in a given MLN. Among others, the possibilistic logic theory could be used to generate explanations for predictions made by the MLN, to gain insight into the data from which the MLN was learned, or to identify errors in the structure or weights of the MLN. Taking this last idea one step further, our aim for future work is to study methods for repairing a given MLN, based on the mistakes that have thus been identified.

### Acknowledgements

We would like to thank the anonymous reviewers for their helpful comments. This work has been supported by a grant from the Leverhulme Trust (RPG-2014-164). JD is partially supported by the Research Fund KU Leuven (OT/11/051), EU FP7 Marie Curie Career Integration Grant (294068) and FWO-Vlaanderen(G.0356.12).



## References

- [1] S. Benferhat, D. Dubois, S. Lagrue, and H. Prade. A big-stepped probability approach for discovering default rules. *Int. Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 11:1–14, 2003.
- [2] S. Benferhat, D. Dubois, and H. Prade. Nonmonotonic reasoning, conditional objects and possibility theory. *Artificial Intelligence*, 92(1-2):259–276, 1997.
- [3] D. L. Berre and A. Parrain. The SAT4J library, release 2.2. *Journal on Satisfiability, Boolean Modeling and Computation*, 7:50–64, 2010.
- [4] C. Cayrol and M.-C. Lagasquie-Schiex. On the complexity of non-monotonic entailment in syntax-based approaches. In *Proceedings of the 11th ECAI Workshop on Algorithms, Complexity and Commonsense Reasoning*, 1994.
- [5] M. W. Craven and J. W. Shavlik. Extracting tree-structured representations of trained networks. In D. Touretzky, M. Mozer, and M. Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8, pages 24–30. MIT Press, 1996.
- [6] D. Dubois. Possibility theory and statistical reasoning. *Computational statistics & data analysis*, 51(1):47–69, 2006.
- [7] D. Dubois, H. Fargier, and H. Prade. Ordinal and probabilistic representations of acceptance. *Journal of Artificial Intelligence Research*, 22:23–56, 2004.
- [8] D. Dubois, J. Lang, and H. Prade. Automated reasoning using possibilistic logic: semantics, belief revision, and variable certainty weights. *IEEE Trans. on Knowledge and Data Engineering*, 6(1):64–71, 1994.
- [9] D. Dubois, J. Lang, and H. Prade. Possibilistic logic. In D. N. D. Gabbay, C. Hogger J. Robinson, editor, *Handbook of Logic in Artificial Intelligence and Logic Programming*, volume 3, pages 439–513. Oxford University Press, 1994.
- [10] D. Dubois and H. Prade. On several representations of an uncertain body of evidence. In M. Gupta and E. Sanchez, editors, *Fuzzy Information and Decision Processes*, pages 167–181. North-Holland, 1982.
- [11] D. Dubois and H. Prade. Fuzzy sets, probability and measurement. *European Journal of Operational Research*, 40(2):135–154, 1989.
- [12] D. Dubois and H. Prade. Possibility theory: qualitative and quantitative aspects. In D. Gabbay and P. Smets, editors, *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, volume 1, pages 169–226. Kluwer Academic, 1998.
- [13] F. Dupin de Saint-Cyr, J. Lang, and T. Schiex. Penalty logic and its link with Dempster-Shafer theory. In *Proc. of the 10th International Conference on Uncertainty in Artificial Intelligence*, pages 204–211, 1994.
- [14] H. Geffner and J. Pearl. Conditional entailment: Bridging two approaches to default reasoning. *Artificial Intelligence*, 53(2):209–244, 1992.
- [15] S. Kraus, D. Lehmann, and M. Magidor. Nonmonotonic reasoning, preferential models and cumulative logics. *Artificial Intelligence*, 44(1-2):167–207, 1990.
- [16] O. Kuželka and F. Železný. A restarted strategy for efficient subsumption testing. *Fundamenta Informaticae*, 89(1):95–109, 2008.
- [17] J. Lang. Possibilistic logic: complexity and algorithms. In J. Kohlas and S. Moral, editors, *Algorithms for Uncertainty and Defeasible Reasoning*, volume 5 of *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, pages 179–220. Kluwer Academic Publishers, 2001.
- [18] J. Noessner, M. Niepert, and H. Stuckenschmidt. RockIt: Exploiting parallelism and symmetry for map inference in statistical relational models. In *Proc. of the 27th Conf. on Artificial Intelligence (AAAI)*, 2013.
- [19] J. Pearl. System Z: A natural ordering of defaults with tractable applications to nonmonotonic reasoning. In *Proc. of the 3rd Conference on Theoretical Aspects of Reasoning about Knowledge*, pages 121–135, 1990.
- [20] D. Poole. First-order probabilistic inference. In *Proceedings of IJCAI*, volume 18, pages 985–991, 2003.
- [21] D. Popovic, A. Sifrim, Y. Moreau, and B. D. Moor. eXtasy simplified - towards opening the black box. In *Proc. of the IEEE International Conf. on Bioinformatics and Biomedicine (BIBM)*, pages 24–28, 2013.
- [22] M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62(1-2):107–136, 2006.
- [23] S. Riedel. Improving the accuracy and efficiency of MAP inference for Markov logic. In *Proc. of the 24th Conference on Uncertainty in Artificial Intelligence*.
- [24] M. Serrurier and H. Prade. Introducing possibilistic logic in ILP for dealing with exceptions. *Artificial Intelligence*, 171(16–17):939–950, 2007.
- [25] G. Shafer. *A mathematical theory of evidence*. Princeton University Press, 1976.
- [26] M. Thimm. Coherence and compatibility of markov logic networks. In *Proc. of the 21st European Conference on Artificial Intelligence*, pages 891–896, 2014.
- [27] L. Zadeh. Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, 1:3–28, 1978.

---

# On the Computability of AIXI

---

**Jan Leike**

Australian National University  
jan.leike@anu.edu.au

**Marcus Hutter**

Australian National University  
marcus.hutter@anu.edu.au

## Abstract

How could we solve the machine learning and the artificial intelligence problem if we had infinite computation? Solomonoff induction and the reinforcement learning agent AIXI are proposed answers to this question. Both are known to be incomputable. In this paper, we quantify this using the arithmetical hierarchy, and prove upper and corresponding lower bounds for incomputability. We show that AIXI is not limit computable, thus it cannot be approximated using finite computation. Our main result is a limit-computable  $\varepsilon$ -optimal version of AIXI with infinite horizon that maximizes expected rewards.

**Keywords.** AIXI, Solomonoff induction, general reinforcement learning, computability, complexity, arithmetical hierarchy, universal Turing machine.

## 1 INTRODUCTION

Given infinite computation power, many traditional AI problems become trivial: playing chess, go, or backgammon can be solved by exhaustive expansion of the game tree. Yet other problems seem difficult still; for example, predicting the stock market, driving a car, or babysitting your nephew. How can we solve these problems in theory? A proposed answer to this question is the agent AIXI [Hut00, Hut05]. As a *reinforcement learning agent*, its goal is to maximize cumulative (discounted) rewards obtained from the environment [SB98].

The basis of AIXI is Solomonoff's theory of learning [Sol64, Sol78, LV08], also called *Solomonoff induction*. It arguably solves the induction problem [RH11]: for data drawn from a computable measure  $\mu$ , Solomonoff induction will converge to the correct belief about any hypothesis [BD62, RH11]. Moreover, convergence is extremely fast in the sense that Solomonoff induction will make a total of at most  $E + O(\sqrt{E})$  errors when predicting

the next data points, where  $E$  is the number of errors of the informed predictor that knows  $\mu$  [Hut01]. While learning the environment according to Solomonoff's theory, AIXI selects actions by running an expectimax-search for maximum cumulative discounted rewards. It is clear that AIXI can only serve as an ideal, yet recently it has inspired some impressive applications [VNH<sup>+</sup>11].

Both Solomonoff induction and AIXI are known to be incomputable. But not all incomputabilities are equal. The *arithmetical hierarchy* specifies different levels of computability based on *oracle machines*: each level in the arithmetical hierarchy is computed by a Turing machine which may query a halting oracle for the respective lower level.

We posit that any ideal for a 'perfect agent' needs to be *limit computable* ( $\Delta_2^0$ ). The class of limit computable functions is the class of functions that admit an *anytime algorithm*. It is the highest level of the arithmetical hierarchy which can be approximated using a regular Turing machine. If this criterion is not met, our model would be useless to guide practical research.

For MDPs, planning is already P-complete for finite and infinite horizons [PT87]. In POMDPs, planning is undecidable [MHC99, MHC03]. The existence of a policy whose expected value exceeds a given threshold is PSPACE-complete [MGLA00], even for purely epistemic POMDPs in which actions do not change the hidden state [SLR07]. In this paper we derive hardness results for planning in general semicomputable environments; this environment class is even more general than POMDPs. We show that finding an optimal policy is  $\Pi_2^0$ -hard and finding an  $\varepsilon$ -optimal policy is undecidable.

Moreover, we show that by default, AIXI is not limit computable. The reason is twofold: First, when picking the next action, two or more actions might have the same value (expected future rewards). The choice between them is easy, but determining whether such a tie exists is difficult. Second, in case of an infinite horizon (using discounting), the iterative definition of the value function [Hut05, Def. 5.30] conditions on surviving forever. The first problem

| Model          | $\gamma$ | Optimal                        | $\varepsilon$ -Optimal         |
|----------------|----------|--------------------------------|--------------------------------|
| Iterative AINU | DC       | $\Delta_4^0, \Sigma_3^0$ -hard | $\Delta_3^0, \Pi_2^0$ -hard    |
|                | LT       | $\Delta_3^0, \Pi_2^0$ -hard    | $\Delta_2^0, \Sigma_1^0$ -hard |
| Iterative AIXI | DC       | $\Delta_4^0, \Pi_2^0$ -hard    | $\Delta_3^0, \Pi_2^0$ -hard    |
|                | LT       | $\Delta_3^0, \Sigma_1^0$ -hard | $\Delta_2^0, \Sigma_1^0$ -hard |
| Iterative AIMU | DC       | $\Delta_2^0$                   | $\Delta_1^0$                   |
|                | LT       | $\Delta_2^0$                   | $\Delta_1^0$                   |
| Recursive AINU | DC       | $\Delta_3^0, \Pi_2^0$ -hard    | $\Delta_2^0, \Sigma_1^0$ -hard |
|                | LT       | $\Delta_3^0, \Pi_2^0$ -hard    | $\Delta_2^0, \Sigma_1^0$ -hard |
| Recursive AIXI | DC       | $\Delta_3^0, \Sigma_1^0$ -hard | $\Delta_2^0, \Sigma_1^0$ -hard |
|                | LT       | $\Delta_3^0, \Sigma_1^0$ -hard | $\Delta_2^0, \Sigma_1^0$ -hard |
| Recursive AIMU | DC       | $\Delta_2^0$                   | $\Delta_1^0$                   |
|                | LT       | $\Delta_2^0$                   | $\Delta_1^0$                   |

Table 1: Computability results for different agent models derived in Section 3. DC means general discounting, a lower semicomputable discount function  $\gamma$ ; LT means finite lifetime, undiscounted rewards up to a fixed lifetime  $m$ . Hardness results for AIXI are with respect to a specific universal Turing machine; hardness results for AINU are with respect to a specific environment  $\nu \in \mathcal{M}$ .

can be circumvented by settling for an  $\varepsilon$ -optimal agent. We show that the second problem can be solved by using the recursive instead of the iterative definition of the value function. With this we get a limit-computable agent with infinite horizon. Table 1 and Table 3 summarize our computability results.

## 2 PRELIMINARIES

### 2.1 THE ARITHMETICAL HIERARCHY

A set  $A \subseteq \mathbb{N}$  is  $\Sigma_n^0$  iff there is a computable relation  $S$  such that

$$k \in A \iff \exists k_1 \forall k_2 \dots Q_n k_n S(k, k_1, \dots, k_n) \quad (1)$$

where  $Q_n = \forall$  if  $n$  is even,  $Q_n = \exists$  if  $n$  is odd [Nie09, Def. 1.4.10]. A set  $A \subseteq \mathbb{N}$  is  $\Pi_n^0$  iff its complement  $\mathbb{N} \setminus A$  is  $\Sigma_n^0$ . We call the formula on the right hand side of (1) a  $\Sigma_n^0$ -formula, its negation is called  $\Pi_n^0$ -formula. It can be shown that we can add any bounded quantifiers and duplicate quantifiers of the same type without changing the classification of  $A$ . The set  $A$  is  $\Delta_n^0$  iff  $A$  is  $\Sigma_n^0$  and  $A$  is  $\Pi_n^0$ . We get that  $\Sigma_1^0$  as the class of recursively enumerable sets,  $\Pi_1^0$  as the class of co-recursively enumerable sets and  $\Delta_1^0$  as the class of recursive sets.

We say the set  $A \subseteq \mathbb{N}$  is  $\Sigma_n^0$ -hard ( $\Pi_n^0$ -hard,  $\Delta_n^0$ -hard) iff for any set  $B \in \Sigma_n^0$  ( $B \in \Pi_n^0$ ,  $B \in \Delta_n^0$ ),  $B$  is many-one reducible to  $A$ , i.e., there is a computable function  $f$  such that  $k \in B \leftrightarrow f(k) \in A$  [Nie09, Def. 1.2.1]. We get  $\Sigma_n^0 \subseteq$

$\Delta_{n+1}^0 \subseteq \Sigma_{n+1}^0 \subseteq \dots$  and  $\Pi_n^0 \subseteq \Delta_{n+1}^0 \subseteq \Pi_{n+1}^0 \subseteq \dots$ . This hierarchy of subsets of natural numbers is known as the *arithmetical hierarchy*.

By Post's Theorem [Nie09, Thm. 1.4.13], a set is  $\Sigma_n^0$  if and only if it is recursively enumerable on an oracle machine with an oracle for a  $\Sigma_{n-1}^0$ -complete set.

### 2.2 STRINGS

Let  $\mathcal{X}$  be some finite set called *alphabet*. The set  $\mathcal{X}^* := \bigcup_{n=0}^{\infty} \mathcal{X}^n$  is the set of all finite strings over the alphabet  $\mathcal{X}$ , the set  $\mathcal{X}^\infty$  is the set of all infinite strings over the alphabet  $\mathcal{X}$ , and the set  $\mathcal{X}^\# := \mathcal{X}^* \cup \mathcal{X}^\infty$  is their union. The empty string is denoted by  $\epsilon$ , not to be confused with the small positive real number  $\varepsilon$ . Given a string  $x \in \mathcal{X}^*$ , we denote its length by  $|x|$ . For a (finite or infinite) string  $x$  of length  $\geq k$ , we denote with  $x_{1:k}$  the first  $k$  characters of  $x$ , and with  $x_{<k}$  the first  $k-1$  characters of  $x$ . The notation  $x_{1:\infty}$  stresses that  $x$  is an infinite string. We write  $x \sqsubseteq y$  iff  $x$  is a prefix of  $y$ , i.e.,  $x = y_{1:|x|}$ .

### 2.3 COMPUTABILITY OF REAL-VALUED FUNCTIONS

We fix some encoding of rational numbers into binary strings and an encoding of binary strings into natural numbers. From now on, this encoding will be done implicitly wherever necessary.

**Definition 1** ( $\Sigma_n^0$ -,  $\Pi_n^0$ -,  $\Delta_n^0$ -computable). A function  $f : \mathcal{X}^* \rightarrow \mathbb{R}$  is called  $\Sigma_n^0$ -computable ( $\Pi_n^0$ -computable,  $\Delta_n^0$ -computable) iff the set  $\{(x, q) \in \mathcal{X}^* \times \mathbb{Q} \mid f(x) > q\}$  is  $\Sigma_n^0$  ( $\Pi_n^0$ ,  $\Delta_n^0$ ).

A  $\Delta_1^0$ -computable function is called *computable*, a  $\Sigma_1^0$ -computable function is called *lower semicomputable*, and a  $\Pi_1^0$ -computable function is called *upper semicomputable*. A  $\Delta_2^0$ -computable function  $f$  is called *limit computable*, because there is a computable function  $\phi$  such that

$$\lim_{k \rightarrow \infty} \phi(x, k) = f(x).$$

The program  $\phi$  that limit computes  $f$  can be thought of as an *anytime algorithm* for  $f$ : we can stop  $\phi$  at any time  $k$  and get a preliminary answer. If the program  $\phi$  ran long enough (which we do not know), this preliminary answer will be close to the correct one.

Limit-computable sets are the highest level in the arithmetical hierarchy that can be approached by a regular Turing machine. Above limit-computable sets we necessarily need some form of halting oracle. See Table 2 for the definition of lower/upper semicomputable and limit-computable functions in terms of the arithmetical hierarchy.

**Lemma 2** (Computability of Arithmetical Operations). *Let  $n > 0$  and let  $f, g : \mathcal{X}^* \rightarrow \mathbb{R}$  be two  $\Delta_n^0$ -computable functions. Then*

|                                 | $f_>$        | $f_<$        |
|---------------------------------|--------------|--------------|
| $f$ is computable               | $\Delta_1^0$ | $\Delta_1^0$ |
| $f$ is lower semicomputable     | $\Sigma_1^0$ | $\Pi_1^0$    |
| $f$ is upper semicomputable     | $\Pi_1^0$    | $\Sigma_1^0$ |
| $f$ is limit computable         | $\Delta_2^0$ | $\Delta_2^0$ |
| $f$ is $\Delta_n^0$ -computable | $\Delta_n^0$ | $\Delta_n^0$ |
| $f$ is $\Sigma_n^0$ -computable | $\Sigma_n^0$ | $\Pi_n^0$    |
| $f$ is $\Pi_n^0$ -computable    | $\Pi_n^0$    | $\Sigma_n^0$ |

Table 2: Connection between the computability of real-valued functions and the arithmetical hierarchy. We use the shorthand  $f_> := \{(x, q) \mid f(x) > q\}$  and  $f_< := \{(x, q) \mid f(x) < q\}$ .

- (i)  $\{(x, y) \mid f(x) > g(y)\}$  is  $\Sigma_n^0$ ,
- (ii)  $\{(x, y) \mid f(x) \leq g(y)\}$  is  $\Pi_n^0$ ,
- (iii)  $f + g$ ,  $f - g$ , and  $f \cdot g$  are  $\Delta_n^0$ -computable, and
- (iv)  $f/g$  is  $\Delta_n^0$ -computable if  $g(x) \neq 0$  for all  $x$ .

## 2.4 ALGORITHMIC INFORMATION THEORY

A *semimeasure* over the alphabet  $\mathcal{X}$  is a function  $\nu : \mathcal{X}^* \rightarrow [0, 1]$  such that (i)  $\nu(\epsilon) \leq 1$ , and (ii)  $\nu(x) \geq \sum_{a \in \mathcal{X}} \nu(xa)$  for all  $x \in \mathcal{X}^*$ . A semimeasure is called (probability) *measure* iff for all  $x$  equalities hold in (i) and (ii). *Solomonoff's prior*  $M$  [Sol64] assigns to a string  $x$  the probability that the reference universal monotone Turing machine  $U$  [LV08, Ch. 4.5.2] computes a string starting with  $x$  when fed with uniformly random bits as input. The *measure mixture*  $\bar{M}$  [Gá83, p. 74] removes the contribution of programs that do not compute infinite strings; it is a measure except for a constant factor. Formally,

$$M(x) := \sum_{p: x \sqsubseteq U(p)} 2^{-|p|}, \quad \bar{M}(x) := \lim_{n \rightarrow \infty} \sum_{y \in \mathcal{X}^n} M(xy)$$

Equivalently, the Solomonoff prior  $M$  can be defined as a mixture over all lower semicomputable semimeasures [WSH11]. The function  $M$  is a lower semicomputable semimeasure, but not computable and not a measure [LV08, Lem. 4.5.3]. A semimeasure  $\nu$  can be turned into a measure  $\nu_{\text{norm}}$  using *Solomonoff normalization*:  $\nu_{\text{norm}}(\epsilon) := 1$  and for all  $x \in \mathcal{X}^*$  and  $a \in \mathcal{X}$ ,

$$\nu_{\text{norm}}(xa) := \nu_{\text{norm}}(x) \frac{\nu(xa)}{\sum_{b \in \mathcal{X}} \nu(xb)}. \quad (2)$$

## 2.5 GENERAL REINFORCEMENT LEARNING

In general reinforcement learning the agent interacts with an environment in cycles: at time step  $t$  the agent chooses an *action*  $a_t \in \mathcal{A}$  and receives a *percept*  $e_t = (o_t, r_t) \in \mathcal{E}$  consisting of an *observation*  $o_t \in \mathcal{O}$  and a real-valued

*reward*  $r_t \in \mathbb{R}$ ; the cycle then repeats for  $t + 1$ . A *history* is an element of  $(\mathcal{A} \times \mathcal{E})^*$ . We use  $\mathfrak{x} \in \mathcal{A} \times \mathcal{E}$  to denote one interaction cycle, and  $\mathfrak{x}_{1:t}$  to denote a history of length  $t$ . The goal in reinforcement learning is to maximize total discounted rewards. A *policy* is a function  $\pi : (\mathcal{A} \times \mathcal{E})^* \rightarrow \mathcal{A}$  mapping each history to the action taken after seeing this history.

The environment can be stochastic, but is assumed to be semicomputable. In accordance with the AIXI literature [Hut05], we model environments as lower semicomputable *chronological conditional semimeasures* (LSCCCSs). A *conditional semimeasure*  $\nu$  takes a sequence of actions  $a_{1:t}$  as input and returns a semimeasure  $\nu(\cdot \parallel a_{1:t})$  over  $\mathcal{E}^\#$ . A conditional semimeasure  $\nu$  is *chronological* iff percepts at time  $t$  do not depend on future actions, i.e.,  $\nu(e_{1:t} \parallel a_{1:k}) = \nu(e_{1:t} \parallel a_{1:t})$  for all  $k > t$ . Despite their name, conditional semimeasures do *not* specify conditional probabilities; the environment  $\nu$  is *not* a joint probability distribution on actions and percepts. Here we only care about the computability of the environment  $\nu$ ; for our purposes, chronological conditional semimeasures behave just like semimeasures.

## 2.6 THE UNIVERSAL AGENT AIXI

Our environment class  $\mathcal{M}$  is the class of all LSCCCSs. Typically, Bayesian agents such as AIXI only function well if the true environment is in their hypothesis class. Since the hypothesis class  $\mathcal{M}$  is extremely large, the assumption that it contains the true environment is rather weak. We fix the *universal prior*  $(w_\nu)_{\nu \in \mathcal{M}}$  with  $w_\nu > 0$  for all  $\nu \in \mathcal{M}$  and  $\sum_{\nu \in \mathcal{M}} w_\nu \leq 1$ , given by the reference machine  $U$ . The universal prior  $w$  gives rise to the *universal mixture*  $\xi$ , which is a convex combination of all LSCCCSs  $\mathcal{M}$ :

$$\xi(e_{<t} \parallel a_{<t}) := \sum_{\nu \in \mathcal{M}} w_\nu \nu(e_{<t} \parallel a_{<t})$$

It is analogous to the Solomonoff prior  $M$  but defined for reactive environments. Like  $M$ , the universal mixture  $\xi$  is lower semicomputable [Hut05, Sec. 5.10].

We fix a *discount function*  $\gamma : \mathbb{N} \rightarrow \mathbb{R}$  with  $\gamma_t := \gamma(t) \geq 0$  and  $\sum_{t=1}^\infty \gamma_t < \infty$  and make the following assumptions.

**Assumption 3.** (a) *The discount function  $\gamma$  is lower semicomputable.*

(b) *Rewards are bounded between 0 and 1.*

(c) *The set of actions  $\mathcal{A}$  and the set of percepts  $\mathcal{E}$  are both finite.*

Assumption 3 (b) could be relaxed to bounded rewards because we can rescale rewards  $r \mapsto cr + d$  for any  $c, d \in \mathbb{R}$  without changing optimal policies if the environment  $\nu$  is a measure. However, for our value-related results, we require that rewards are nonnegative.

We define the *discount normalization factor*  $\Gamma_t := \sum_{i=t}^{\infty} \gamma_i$ . There is no requirement that  $\Gamma_t > 0$ . In fact, we use  $\gamma$  for both, AIXI with discounted infinite horizon ( $\Gamma_t > 0$  for all  $t$ ), and AIXI with finite lifetime  $m$ . In the latter case we set

$$\gamma_{\text{LT}m}(t) := \begin{cases} 1 & \text{if } t \leq m \\ 0 & \text{if } t > m. \end{cases}$$

If we knew the true environment  $\nu \in \mathcal{M}$ , we would choose the  $\nu$ -optimal agent known as AINU that maximizes  $\nu$ -expected value (if  $\nu$  is a measure). Since we do not know the true environment, we use the universal mixture  $\xi$  over all environments in  $\mathcal{M}$  instead. This yields the Bayesian agent AIXI: it weighs every environment  $\nu \in \mathcal{M}$  according to its prior probability  $w_\nu$ .

**Definition 4** (Iterative Value Function [Hut05, Def. 5.30]). The *value* of a policy  $\pi$  in an environment  $\nu$  given history  $\mathbf{x}_{<t}$  is

$$V_\nu^\pi(\mathbf{x}_{<t}) := \frac{1}{\Gamma_t} \lim_{m \rightarrow \infty} \sum_{e_{t:m}} R(e_{t:m}) \nu(e_{1:m} \mid e_{<t} \parallel a_{1:m})$$

if  $\Gamma_t > 0$  and  $V_\nu^\pi(\mathbf{x}_{<t}) := 0$  if  $\Gamma_t = 0$  where  $a_i := \pi(e_{<i})$  for all  $i \geq t$  and  $R(e_{t:m}) := \sum_{k=t}^m \gamma_k r_k$ . The *optimal value* is defined as  $V_\nu^*(h) := \sup_\pi V_\nu^\pi(h)$ .

Let  $\mathbf{x}_{<t} \in (\mathcal{A} \times \mathcal{E})^*$  be some history. We extend the value functions  $V_\nu^\pi$  to include initial interactions (in reinforcement learning literature on MDPs these are called  $Q$ -values),  $V_\nu^\pi(\mathbf{x}_{<t} a_t) := V_\nu^{\pi'}(\mathbf{x}_{<t})$  where  $\pi'$  is the policy  $\pi$  except that it takes action  $a_t$  next, i.e.,  $\pi'(\mathbf{x}_{<t}) := a_t$  and  $\pi'(h) := \pi(h)$  for all  $h \neq \mathbf{x}_{<t}$ . We define  $V_\nu^*(\mathbf{x}_{<t} a_t) := \sup_\pi V_\nu^\pi(\mathbf{x}_{<t} a_t)$  analogously.

**Definition 5** (Optimal Policy [Hut05, Def. 5.19 & 5.30]). A policy  $\pi$  is *optimal in environment  $\nu$*  ( $\nu$ -optimal) iff for all histories the policy  $\pi$  attains the optimal value:  $V_\nu^\pi(h) = V_\nu^*(h)$  for all  $h \in (\mathcal{A} \times \mathcal{E})^*$ .

Since the discount function is summable, rewards are bounded (Assumption 3b), and actions and percepts spaces are both finite (Assumption 3c), an optimal policy exists for every environment  $\nu \in \mathcal{M}$  [LH14, Thm. 10]. For a fixed environment  $\nu$ , an explicit expression for the optimal value function is

$$V_\nu^*(\mathbf{x}_{<t}) = \frac{1}{\Gamma_t} \lim_{m \rightarrow \infty} \mathop{\text{m}\max}_{\mathbf{x}_{t:m}} \sum R(e_{t:m}) \nu(e_{1:m} \mid e_{<t} \parallel a_{1:m}), \quad (3)$$

where  $\mathop{\text{m}\max}$  denotes the expectimax operator:

$$\mathop{\text{m}\max}_{\mathbf{x}_{t:m}} := \max_{a_t \in \mathcal{A}} \sum_{e_t \in \mathcal{E}} \dots \max_{a_m \in \mathcal{A}} \sum_{e_m \in \mathcal{E}}$$

For an environment  $\nu \in \mathcal{M}$  (an LSCCCS), AINU is defined as a  $\nu$ -optimal policy  $\pi_\nu^* = \arg \max_\pi V_\nu^\pi(\epsilon)$ . To

|                              | Plain  | Conditional                                      |
|------------------------------|--|--|
| $M$                          | $\Sigma_1^0 \setminus \Delta_1^0$                | $\Delta_2^0 \setminus (\Sigma_1^0 \cup \Pi_1^0)$ |
| $M_{\text{norm}}$            | $\Delta_2^0 \setminus (\Sigma_1^0 \cup \Pi_1^0)$ | $\Delta_2^0 \setminus (\Sigma_1^0 \cup \Pi_1^0)$ |
| $\overline{M}$               | $\Pi_2^0 \setminus \Delta_2^0$                   | $\Delta_3^0 \setminus (\Sigma_2^0 \cup \Pi_2^0)$ |
| $\overline{M}_{\text{norm}}$ | $\Delta_3^0 \setminus (\Sigma_2^0 \cup \Pi_2^0)$ | $\Delta_3^0 \setminus (\Sigma_2^0 \cup \Pi_2^0)$ |

Table 3: The complexity of the set  $\{(x, q) \in \mathcal{X}^* \times \mathbb{Q} \mid f(x) > q\}$  where  $f \in \{M, M_{\text{norm}}, \overline{M}, \overline{M}_{\text{norm}}\}$  is one of the various versions of Solomonoff’s prior. Lower bounds on the complexity of  $\overline{M}$  and  $\overline{M}_{\text{norm}}$  hold only for specific universal Turing machines.

stress that the environment is given by a measure  $\mu \in \mathcal{M}$  (as opposed to a semimeasure), we use AIMU. AIXI is defined as a  $\xi$ -optimal policy  $\pi_\xi^*$  for the universal mixture  $\xi$  [Hut05, Ch. 5]. Since  $\xi \in \mathcal{M}$  and every measure  $\mu \in \mathcal{M}$  is also a semimeasure, both AIMU and AIXI are a special case of AINU. However, AIXI is not a special case of AIMU since the mixture  $\xi$  is not a measure.

Because there can be more than one optimal policy, the definitions of AINU, AIMU and AIXI are not unique. More specifically, a  $\nu$ -optimal policy maps a history  $h$  to

$$\pi_\nu^*(h) := \arg \max_{a \in \mathcal{A}} V_\nu^*(ha). \quad (4)$$

If there are multiple actions  $\alpha, \beta \in \mathcal{A}$  that attain the optimal value,  $V_\nu^*(h\alpha) = V_\nu^*(h\beta)$ , we say there is an *argmax tie*. Which action we settle on in case of a tie (how we break the tie) is irrelevant and can be arbitrary.

### 3 THE COMPLEXITY OF AINU, AIMU, AND AIXI

#### 3.1 THE COMPLEXITY OF SOLOMONOFF INDUCTION

AIXI uses an analogue to Solomonoff’s prior on all possible environments  $\mathcal{M}$ . Therefore we first state computability results for *Solomonoff’s prior*  $M$  and the *measure mixture*  $\overline{M}$  in Table 3 [LH15b]. Notably,  $M$  is lower semicomputable and its conditional is limit computable. However, neither the measure mixture  $\overline{M}$  nor any of its variants are limit computable.

#### 3.2 UPPER BOUNDS

In this section, we derive upper bounds on the computability of AINU, AIMU, and AIXI. Except for Corollary 13, all results in this section apply generally to any LSCCCS  $\nu \in \mathcal{M}$ , hence they apply to AIXI even though they are stated for AINU.

For a fixed lifetime  $m$ , only the first  $m$  interactions matter. There is a finite number of policies that are different for

the first  $m$  interactions, and the optimal policy  $\pi_\xi^*$  can be encoded in a finite number of bits and is thus computable. To make a meaningful statement about the computability of  $\text{AINU}_{\text{LT}}$ , we have to consider it as the function that takes the lifetime  $m$  and outputs a policy  $\pi_\xi^*$  that is optimal in the environment  $\xi$  using the discount function  $\gamma_{\text{LT}m}$ . In contrast, for infinite lifetime discounting we just consider the function  $\pi_\xi^* : (\mathcal{A} \times \mathcal{E})^* \rightarrow \mathcal{A}$ .

In order to position AINU in the arithmetical hierarchy, we need to identify these functions with sets of natural numbers. In both cases, finite and infinite lifetime, we represent these functions as relations over  $\mathbb{N} \times (\mathcal{A} \times \mathcal{E})^* \times \mathcal{A}$  and  $(\mathcal{A} \times \mathcal{E})^* \times \mathcal{A}$  respectively. These relations are easily identified with sets of natural numbers by encoding the tuple with their arguments into one natural number. From now on this translation of policies (and  $m$ ) into sets of natural numbers will be done implicitly wherever necessary.

**Lemma 6** (Policies are in  $\Delta_n^0$ ). *If a policy  $\pi$  is  $\Sigma_n^0$  or  $\Pi_n^0$ , then  $\pi$  is  $\Delta_n^0$ .*

*Proof.* Let  $\varphi$  be a  $\Sigma_n^0$ -formula ( $\Pi_n^0$ -formula) defining  $\pi$ , i.e.,  $\varphi(h, a)$  holds iff  $\pi(h) = a$ . We define the formula  $\varphi'$ ,

$$\varphi'(h, a) := \bigwedge_{a' \in \mathcal{A} \setminus \{a\}} \neg \varphi(h, a').$$

The set of actions  $\mathcal{A}$  is finite, hence  $\varphi'$  is a  $\Pi_n^0$ -formula ( $\Sigma_n^0$ -formula). Moreover,  $\varphi'$  is equivalent to  $\varphi$ .  $\square$

To compute the optimal policy, we need to compute the value function. The following lemma gives an upper bound on the computability of the value function for environments in  $\mathcal{M}$ .

**Lemma 7** (Complexity of  $V_\nu^*$ ). *For every LSCCCS  $\nu \in \mathcal{M}$ , the function  $V_\nu^*$  is  $\Pi_2^0$ -computable. For  $\gamma = \gamma_{\text{LT}m}$  the function  $V_\nu^*$  is  $\Delta_2^0$ -computable.*

*Proof.* Multiplying (3) with  $\Gamma_t \nu(e_{<t} \parallel a_{<t})$  yields  $V_\nu^*(\mathbf{x}_{<t}) > q$  if and only if

$$\lim_{m \rightarrow \infty} \bigvee_{\mathbf{x}_{t:m}} \nu(e_{1:m} \parallel a_{1:m}) R(e_{t:m}) > q \Gamma_t \nu(e_{<t} \parallel a_{<t}). \quad (5)$$

The inequality's right side is lower semicomputable, hence there is a computable function  $\psi$  such that  $\psi(\ell) \nearrow q \Gamma_t \nu(e_{<t} \parallel a_{<t}) =: q'$  for  $\ell \rightarrow \infty$ . For a fixed  $m$ , the left side is also lower semicomputable, therefore there is a computable function  $\phi$  such that  $\phi(m, k) \nearrow \bigvee_{\mathbf{x}_{t:m}} \nu(e_{1:m} \parallel a_{1:m}) R(e_{t:m}) =: f(m)$  for  $k \rightarrow \infty$ . We already know that the limit of  $f(m)$  for  $m \rightarrow \infty$  exists (uniquely), hence we

can write (5) as

$$\begin{aligned} & \lim_{m \rightarrow \infty} f(m) > q' \\ \iff & \forall m_0 \exists m \geq m_0. f(m) > q' \\ \iff & \forall m_0 \exists m \geq m_0 \exists k. \phi(m, k) > q' \\ \iff & \forall \ell \forall m_0 \exists m \geq m_0 \exists k. \phi(m, k) > \psi(\ell), \end{aligned}$$

which is a  $\Pi_2^0$ -formula. In the finite lifetime case where  $m$  is fixed, the value function  $V_\nu^*(\mathbf{x}_{<t})$  is  $\Delta_2^0$ -computable by Lemma 2 (iv), since  $V_\nu^*(\mathbf{x}_{<t}) = f(m)q/q'$ .  $\square$

From the optimal value function  $V_\nu^*$  we get the optimal policy  $\pi_\nu^*$  according to (4). However, in cases where there is more than one optimal action, we have to break an argmax tie. This happens iff  $V_\nu^*(h\alpha) = V_\nu^*(h\beta)$  for two potential actions  $\alpha \neq \beta \in \mathcal{A}$ . This equality test is more difficult than determining which is larger in cases where they are unequal. Thus we get the following upper bound.

**Theorem 8** (Complexity of Optimal Policies). *For any environment  $\nu \in \mathcal{M}$ , if  $V_\nu^*$  is  $\Delta_n^0$ -computable, then there is an optimal policy  $\pi_\nu^*$  for the environment  $\nu$  that is  $\Delta_{n+1}^0$ .*

*Proof.* To break potential ties, we pick an (arbitrary) total order  $\succ$  on  $\mathcal{A}$  that specifies which actions should be preferred in case of a tie. We define

$$\begin{aligned} \pi_\nu(h) = a \iff & \bigwedge_{a': a' \succ a} V_\nu^*(ha) > V_\nu^*(ha') \\ & \wedge \bigwedge_{a': a' \succ a'} V_\nu^*(ha) \geq V_\nu^*(ha'). \end{aligned} \quad (6)$$

Then  $\pi_\nu$  is a  $\nu$ -optimal policy according to (4). By assumption,  $V_\nu^*$  is  $\Delta_n^0$ -computable. By Lemma 2 (i) and (ii)  $V_\nu^*(ha) > V_\nu^*(ha')$  is in  $\Sigma_n^0$  and  $V_\nu^*(ha) \geq V_\nu^*(ha')$  is  $\Pi_n^0$ . Therefore the policy  $\pi_\nu$  defined in (6) is a conjunction of a  $\Sigma_n^0$ -formula and a  $\Pi_n^0$ -formula and thus in  $\Delta_{n+1}^0$ .  $\square$

**Corollary 9** (Complexity of AINU).  *$\text{AINU}_{\text{LT}}$  is  $\Delta_3^0$  and  $\text{AINU}_{\text{DC}}$  is  $\Delta_4^0$  for every environment  $\nu \in \mathcal{M}$ .*

*Proof.* From Lemma 7 and Theorem 8.  $\square$

Usually we do not mind taking slightly suboptimal actions. Therefore actually trying to determine if two actions have the exact same value seems like a waste of resources. In the following, we consider policies that attain a value that is always within some  $\varepsilon > 0$  of the optimal value.

**Definition 10** ( $\varepsilon$ -Optimal Policy). *A policy  $\pi$  is  $\varepsilon$ -optimal in environment  $\nu$  iff  $V_\nu^*(h) - V_\nu^\pi(h) < \varepsilon$  for all histories  $h \in (\mathcal{A} \times \mathcal{E})^*$ .*

**Theorem 11** (Complexity of  $\varepsilon$ -Optimal Policies). *For any environment  $\nu \in \mathcal{M}$ , if  $V_\nu^*$  is  $\Delta_n^0$ -computable, then there is an  $\varepsilon$ -optimal policy  $\pi_\nu^\varepsilon$  for the environment  $\nu$  that is  $\Delta_n^0$ .*

*Proof.* Let  $\varepsilon > 0$  be given. Since the value function  $V_\nu^*(h)$  is  $\Delta_n^0$ -computable, the set  $V_\varepsilon := \{(ha, q) \mid |q - V_\nu^*(ha)| < \varepsilon/2\}$  is in  $\Delta_n^0$  according to Definition 1. Hence we compute the values  $V_\nu^*(ha')$  until we get within  $\varepsilon/2$  for every  $a' \in \mathcal{A}$  and then choose the action with the highest value so far. Formally, let  $\succ$  be an arbitrary total order on  $\mathcal{A}$  that specifies which actions should be preferred in case of a tie. Without loss of generality, we assume  $\varepsilon = 1/k$ , and define  $Q$  to be an  $\varepsilon/2$ -grid on  $[0, 1]$ , i.e.,  $Q := \{0, 1/2k, 2/2k, \dots, 1\}$ . We define

$$\begin{aligned} \pi_\nu^\varepsilon(h) = a &: \iff \\ \exists (q_{a'})_{a' \in \mathcal{A}} \in Q^{\mathcal{A}}. & \bigwedge_{a' \in \mathcal{A}} (ha', q_{a'}) \in V_\varepsilon \\ & \wedge \bigwedge_{a': a' \succ a} q_a > q_{a'} \wedge \bigwedge_{a': a \succ a'} q_a \geq q_{a'} \\ & \wedge \text{the tuple } (q_{a'})_{a' \in \mathcal{A}} \text{ is minimal with} \\ & \text{respect to the lex. ordering on } Q^{\mathcal{A}}. \end{aligned} \quad (7)$$

This makes the choice of  $a$  unique. Moreover,  $Q^{\mathcal{A}}$  is finite since  $\mathcal{A}$  is finite, and hence (7) is a  $\Delta_n^0$ -formula.  $\square$

**Corollary 12** (Complexity of  $\varepsilon$ -Optimal AINU). *For any environment  $\nu \in \mathcal{M}$ , there is an  $\varepsilon$ -optimal policy for  $\text{AINU}_{LT}$  that is  $\Delta_2^0$  and there is an  $\varepsilon$ -optimal policy for  $\text{AINU}_{DC}$  that is  $\Delta_3^0$ .*

*Proof.* From Lemma 7 and Theorem 11.  $\square$

If the environment  $\nu \in \mathcal{M}$  is a measure, i.e.,  $\nu$  assigns zero probability to finite strings, then we get computable  $\varepsilon$ -optimal policies.

**Corollary 13** (Complexity of AIMU). *If the environment  $\mu \in \mathcal{M}$  is a measure and the discount function  $\gamma$  is computable, then  $\text{AIMU}_{LT}$  and  $\text{AIMU}_{DC}$  are limit computable ( $\Delta_2^0$ ), and  $\varepsilon$ -optimal  $\text{AIMU}_{LT}$  and  $\text{AIMU}_{DC}$  are computable ( $\Delta_1^0$ ).*

*Proof.* In the discounted case, we can truncate the limit  $m \rightarrow \infty$  in (3) at the  $\varepsilon/2$ -effective horizon  $m_{\text{eff}} := \min\{k \mid \Gamma_k/\Gamma_t < \varepsilon/2\}$ , since everything after  $m_{\text{eff}}$  can contribute at most  $\varepsilon/2$  to the value function. Any lower semicomputable measure is computable [LV08, Lem. 4.5.1]. Therefore  $V_\mu^*$  as given in (3) is composed only of computable functions, hence it is computable according to Lemma 2. The claim now follows from Theorem 8 and Theorem 11.  $\square$

### 3.3 LOWER BOUNDS

We proceed to show that the bounds from the previous section are the best we can hope for. In environment classes where ties have to be broken,  $\text{AIMU}_{DC}$  has to solve  $\Sigma_3^0$ -hard problems (Theorem 15), and  $\text{AIMU}_{LT}$  has to solve

$\Pi_2^0$ -hard problems (Theorem 16). These lower bounds are stated for particular environments  $\nu \in \mathcal{M}$ .

We also construct universal mixtures that yield bounds on  $\varepsilon$ -optimal policies. In the finite lifetime case, there is an  $\varepsilon$ -optimal  $\text{AIXI}_{LT}$  that solves  $\Sigma_1^0$ -hard problems (Theorem 17), and for general discounting, there is an  $\varepsilon$ -optimal  $\text{AIXI}_{DC}$  that solves  $\Pi_2^0$ -hard problems (Theorem 18). For arbitrary universal mixtures, we prove the following weaker statement that only guarantees incomputability.

**Theorem 14** (No AIXI is computable).  *$\text{AIXI}_{LT}$  and  $\text{AIXI}_{DC}$  are not computable for any universal Turing machine  $U$ .*

This theorem follows from the incomputability of Solomonoff induction. Since AIXI uses an analogue of Solomonoff's prior for learning, it succeeds to predict the environment's behavior for its own policy [Hut05, Thm. 5.31]. If AIXI were computable, then there would be computable environments more powerful than AIXI: they can simulate AIXI and anticipate its prediction, which leads to a contradiction.

*Proof.* Assume there is a computable policy  $\pi_\xi^*$  that is optimal in  $\xi$ . We define a deterministic environment  $\mu$ , the adversarial environment to  $\pi_\xi^*$ . The environment  $\mu$  gives rewards 0 as long as the agent follows the policy  $\pi_\xi^*$ , and rewards 1 once the agent deviates. Formally, we ignore observations by setting  $\mathcal{O} := \{0\}$ , and define

$$\begin{aligned} \mu(r_{1:t} \parallel a_{1:t}) &:= \\ \begin{cases} 1 & \text{if } \forall k \leq t. a_k = \pi_\xi^*((ar)_{<k}) \text{ and } r_k = 0 \\ 1 & \text{if } \forall k \leq t. r_k = \mathbb{1}_{k \geq i} \\ & \text{where } i := \min\{j \mid a_j \neq \pi_\xi^*((ar)_{<j})\} \\ 0 & \text{otherwise.} \end{cases} \end{aligned}$$

The environment  $\mu$  is computable because the policy  $\pi_\xi^*$  was assumed to be computable. Suppose  $\pi_\xi^*$  acts in  $\mu$ , then by [Hut05, Thm. 5.36], AIXI learns to predict perfectly on policy:

$$V_\xi^*(\mathbf{x}_{<t}) = V_\xi^{\pi_\xi^*}(\mathbf{x}_{<t}) \rightarrow V_\mu^{\pi_\xi^*}(\mathbf{x}_{<t}) = 0 \text{ as } t \rightarrow \infty,$$

since both  $\pi_\xi^*$  and  $\mu$  are deterministic. Therefore we find a  $t$  large enough such that  $V_\xi^*(\mathbf{x}_{<t}) < w_\mu$  (in the finite lifetime case we choose  $m > t$ ) where  $\mathbf{x}_{<t}$  is the interaction history of  $\pi_\xi^*$  in  $\mu$ . A policy  $\pi$  with  $\pi(\mathbf{x}_{<t}) \neq \pi_\xi^*(\mathbf{x}_{<t})$ , gets a reward of 1 in environment  $\mu$  for all time steps after  $t$ , hence  $V_\mu^\pi(\mathbf{x}_{<t}) = 1$ . With linearity of  $V_\xi^\pi(\mathbf{x}_{<t})$  in  $\xi$  [Hut05, Thm. 5.31],

$$V_\xi^\pi(\mathbf{x}_{<t}) \geq w_\mu \frac{\mu(e_{1:t} \parallel a_{1:t})}{\xi(e_{1:t} \parallel a_{1:t})} V_\mu^\pi(\mathbf{x}_{<t}) \geq w_\mu,$$

since  $\mu(e_{1:t} \parallel a_{1:t}) = 1$  ( $\mu$  is deterministic),  $V_\mu^\pi(\mathbf{x}_{<t}) = 1$ , and  $\xi(e_{1:t} \parallel a_{1:t}) \leq 1$ . Now we get a contradiction:

$$w_\mu > V_\xi^*(\mathbf{x}_{<t}) = \max_{\pi'} V_\xi^{\pi'}(\mathbf{x}_{<t}) \geq V_\xi^\pi(\mathbf{x}_{<t}) \geq w_\mu \quad \square$$

For the remainder of this section, we fix the action space to be  $\mathcal{A} := \{\alpha, \beta\}$  with action  $\alpha$  favored in ties. The percept space is fixed to a tuple of binary observations and rewards,  $\mathcal{E} := \mathcal{O} \times \{0, 1\}$  with  $\mathcal{O} := \{0, 1\}$ .

**Theorem 15** (AINU<sub>DC</sub> is  $\Sigma_3^0$ -hard). *If  $\Gamma_t > 0$  for all  $t$ , there is an environment  $\nu \in \mathcal{M}$  such that AINU<sub>DC</sub> is  $\Sigma_3^0$ -hard.*

*Proof.* Let  $A$  be any  $\Sigma_3^0$  set, then there is a computable relation  $S$  such that

$$n \in A \iff \exists i \forall t \exists k S(n, i, t, k). \quad (8)$$

We define a class of environments  $\mathcal{M}' = \{\rho_0, \rho_1, \dots\} \subseteq \mathcal{M}$  where each environment  $\rho_i$  is defined by

$$\rho_i((or)_{1:t} \parallel a_{1:t}) := \begin{cases} 2^{-t}, & \text{if } o_{1:t} = 1^t \text{ and } \forall t' \leq t. r_{t'} = 0 \\ 2^{-n-1}, & \text{if } \exists n. 1^n 0 \sqsubseteq o_{1:t} \sqsubseteq 1^n 0^\infty \text{ and } a_{n+2} = \alpha \\ & \text{and } \forall t' \leq t. r_{t'} = 0 \\ 2^{-n-1}, & \text{if } \exists n. 1^n 0 \sqsubseteq o_{1:t} \sqsubseteq 1^n 0^\infty \text{ and } a_{n+2} = \beta \\ & \text{and } \forall t' \leq t. r_{t'} = \mathbb{1}_{t' > n+1} \\ & \text{and } \forall t' \leq t \exists k S(n, i, t', k) \\ 0, & \text{otherwise.} \end{cases}$$

Every  $\rho_i$  is a chronological conditional semimeasure by definition, so  $\mathcal{M}' \subseteq \mathcal{M}$ . Furthermore, every  $\rho_i$  is lower semicomputable since  $S$  is computable.

We define our environment  $\nu$  as a mixture over  $\mathcal{M}'$ ,

$$\nu := \sum_{i \in \mathbb{N}} 2^{-i-1} \rho_i;$$

the choice of the weights on the environments  $\rho_i$  is arbitrary but positive. Let  $\pi_\nu^*$  be an optimal policy for the environment  $\nu$  and recall that the action  $\alpha$  is preferred in ties. We claim that for the  $\nu$ -optimal policy  $\pi_\nu^*$ ,

$$n \in A \iff \pi_\nu^*(1^n 0) = \beta. \quad (9)$$

This enables us to decide whether  $n \in A$  given the policy  $\pi_\nu^*$ , hence proving (9) concludes this proof.

Let  $n, i \in \mathbb{N}$  be given, and suppose we are in environment  $i$  and observe  $1^n 0$ . Taking action  $\alpha$  next yields rewards 0 forever; taking action  $\beta$  next yields a reward of 1 for those time steps  $t \geq n + 2$  for which  $\forall t' \leq t \exists k S(n, i, t', k)$ , after that the semimeasure assigns probability 0 to all next observations. Therefore, if for some  $t_0$  there is no  $k$  such that  $S(n, i, t_0, k)$ , then  $\rho_i(e_{1:t_0} \parallel \dots \beta \dots) = 0$ , and hence

$$V_{\rho_i}^*(1^n 0 \beta) = 0 = V_{\rho_i}^*(1^n 0 \alpha),$$

and otherwise  $\rho_i$  yields reward 1 for every time step after  $n + 1$ , therefore

$$V_{\rho_i}^*(1^n 0 \beta) = \Gamma_{n+2} > 0 = V_{\rho_i}^*(1^n 0 \alpha)$$

(omitting the first  $n + 1$  actions and rewards in the argument of the value function). We can now show (9): By (8),  $n \in A$  if and only if there is an  $i$  such that for all  $t$  there is a  $k$  such that  $S(n, i, t, k)$ , which happens if and only if there is an  $i \in \mathbb{N}$  such that  $V_{\rho_i}^*(1^n 0 \beta) > 0$ , which is equivalent to  $V_\nu^*(1^n 0 \beta) > 0$ , which in turn is equivalent to  $\pi_\nu^*(1^n 0) = \beta$  since  $V_\nu^*(1^n 0 \alpha) = 0$  and action  $\alpha$  is favored in ties.  $\square$

**Theorem 16** (AINU<sub>LT</sub> is  $\Pi_2^0$ -hard). *There is an environment  $\nu \in \mathcal{M}$  such that AINU<sub>LT</sub> is  $\Pi_2^0$ -hard.*

The proof of Theorem 16 is analogous to the proof of Theorem 15. The notable difference is that we replace  $\forall t' \leq t \exists k S(n, i, t', k)$  with  $\exists k S(n, i, k)$ . Moreover, we swap actions  $\alpha$  and  $\beta$ : action  $\alpha$  ‘checks’ the relation  $S$  and action  $\beta$  gives a sure reward of 1.

**Theorem 17** (Some  $\varepsilon$ -optimal AIXI<sub>LT</sub> are  $\Sigma_1^0$ -hard). *There is a universal Turing machine  $U'$  and an  $\varepsilon > 0$  such that any  $\varepsilon$ -optimal policy for AIXI<sub>LT</sub> is  $\Sigma_1^0$ -hard.*

*Proof.* Let  $\xi$  denote the universal mixture derived from the reference universal monotone Turing machine  $U$ . Let  $A$  be a  $\Sigma_1^0$ -set and  $S$  computable relation such that  $n + 1 \in A$  iff  $\exists k S(n, k)$ . We define the environment

$$\nu((or)_{1:t} \parallel a_{1:t}) := \begin{cases} \xi((or)_{1:n} \parallel a_{1:n}), & \text{if } \exists n. o_{1:n} = 1^{n-1} 0 \\ & \text{and } a_n = \alpha \\ & \text{and } \forall t' > n. e_{t'} = (0, \frac{1}{2}) \\ \xi((or)_{1:n} \parallel a_{1:n}), & \text{if } \exists n. o_{1:n} = 1^{n-1} 0 \\ & \text{and } a_n = \beta \\ & \text{and } \forall t' > n. e_{t'} = (0, 1) \\ & \text{and } \exists k S(n-1, k). \\ \xi((or)_{1:t} \parallel a_{1:t}), & \text{if } \nexists n. o_{1:n} = 1^{n-1} 0 \\ 0, & \text{otherwise.} \end{cases}$$

The environment  $\nu$  mimics the universal environment  $\xi$  until the observation history is  $1^{n-1} 0$ . Taking the action  $\alpha$  next gives rewards  $1/2$  forever. Taking the action  $\beta$  next gives rewards 1 forever if  $n \in A$ , otherwise the environment  $\nu$  ends at some future time step. Therefore we want to take action  $\beta$  if and only if  $n \in A$ . We have that  $\nu$  is an LSCCCS since  $\xi$  is an LSCCCS and  $S$  is computable.

We define the universal lower semicomputable semimeasure  $\xi' := \frac{1}{2}\nu + \frac{1}{8}\xi$ . Choose  $\varepsilon := 1/9$ . Let  $n \in A$  be given and define the lifetime  $m := n + 1$ . Let  $h \in (\mathcal{A} \times \mathcal{E})^n$  be any history with observations  $o_{1:n} = 1^{n-1} 0$ . Since  $\nu(1^{n-1} 0 \mid a_{1:n}) = \xi(1^{n-1} 0 \mid a_{1:n})$  by definition, the posterior weights of  $\nu$  and  $\xi$  in  $\xi'$  are equal to the prior weights, analogously to [LH15a, Thm. 7]. In the following, we use the linearity of  $V_\rho^{\pi_{\xi'}}^*$  in  $\rho$  [Hut05, Thm. 5.21], and the fact that values are bounded between 0 and 1. If there is a  $k$



such that  $S(n-1, k)$ ,

$$\begin{aligned} & V_{\xi'}^*(h\beta) - V_{\xi'}^*(h\alpha) \\ &= \frac{1}{2}V_{\nu}^{\pi_{\xi'}^*}(h\beta) - \frac{1}{2}V_{\nu}^{\pi_{\xi'}^*}(h\alpha) + \frac{1}{8}V_{\xi}^{\pi_{\xi'}^*}(h\beta) - \frac{1}{8}V_{\xi}^{\pi_{\xi'}^*}(h\alpha) \\ &\geq \frac{1}{2} - \frac{1}{4} + 0 - \frac{1}{8} = \frac{1}{8}, \end{aligned}$$

and similarly if there is no  $k$  such that  $S(n-1, k)$ , then

$$\begin{aligned} & V_{\xi'}^*(h\alpha) - V_{\xi'}^*(h\beta) \\ &= \frac{1}{2}V_{\nu}^{\pi_{\xi'}^*}(h\alpha) - \frac{1}{2}V_{\nu}^{\pi_{\xi'}^*}(h\beta) + \frac{1}{8}V_{\xi}^{\pi_{\xi'}^*}(h\alpha) - \frac{1}{8}V_{\xi}^{\pi_{\xi'}^*}(h\beta) \\ &\geq \frac{1}{4} - 0 + 0 - \frac{1}{8} = \frac{1}{8}. \end{aligned}$$

In both cases  $|V_{\xi'}^*(h\beta) - V_{\xi'}^*(h\alpha)| > 1/9$ . Hence we pick  $\varepsilon := 1/9$  and get for every  $\varepsilon$ -optimal policy  $\pi_{\xi'}^{\varepsilon}$ , that  $\pi_{\xi'}^{\varepsilon}(h) = \beta$  if and only if  $n \in A$ .  $\square$

**Theorem 18** (Some  $\varepsilon$ -optimal  $\text{AIXI}_{\text{DC}}$  are  $\Pi_2^0$ -hard). *There is a universal Turing machine  $U'$  and an  $\varepsilon > 0$  such that any  $\varepsilon$ -optimal policy for  $\text{AIXI}_{\text{DC}}$  is  $\Pi_2^0$ -hard.*

The proof of Theorem 18 is analogous to the proof of Theorem 17 except that we choose  $\forall m' \leq m \exists k S(x, m, k)$  as a condition for reward 1 after playing action  $\beta$ .

## 4 ITERATIVE VS. RECURSIVE AINU

Generally, our environment  $\nu \in \mathcal{M}$  is only a semimeasure and not a measure. I.e., there is a history  $\mathbf{x}_{<t}$  such that

$$1 > \sum_{e_t \in \mathcal{E}} \nu(e_t \mid e_{<t} \parallel a_{1:t}).$$

In such cases, with positive probability the environment  $\nu$  does not produce a new percept  $e_t$ . If this occurs, we shall use the informal interpretation that the environment  $\nu$  *ended*, but our formal argument does not rely on this interpretation.

The following proposition shows that for a semimeasure  $\nu \in \mathcal{M}$  that is not a measure, the iterative definition of AINU does not maximize  $\nu$ -expected rewards. Recall that  $\gamma_1$  states the discount of the first reward. In the following, we assume without loss of generality that  $\gamma_1 > 0$ , i.e., we are not indifferent about the reward received in time step 1.

**Proposition 19** (Iterative AINU is not a  $\nu$ -Expected Rewards Maximizer). *For any  $\varepsilon > 0$  there is an environment  $\nu \in \mathcal{M}$  that is not a measure and a policy  $\pi$  that receives a total of  $\gamma_1$  rewards in  $\nu$ , but AINU receives only  $\varepsilon\gamma_1$  rewards in  $\nu$ .*

Informally, the environment  $\nu$  is defined as follows. In the first time step, the agent chooses between the two actions  $\alpha$  and  $\beta$ . Taking action  $\alpha$  gives a reward of 1, and subsequently the environment ends. Action  $\beta$  gives a reward of  $\varepsilon$ , but the environment continues forever. There are no

other rewards in this environment. From the perspective of  $\nu$ -expected reward maximization, it is better to take action  $\alpha$ , however AINU takes action  $\beta$ .

*Proof.* Let  $\varepsilon > 0$ . We ignore observations and set  $\mathcal{E} := \{0, \varepsilon, 1\}$ ,  $\mathcal{A} := \{\alpha, \beta\}$ . The environment  $\nu$  is formally defined by

$$\nu(r_{1:t} \parallel a_{1:t}) := \begin{cases} 1 & \text{if } a_1 = \alpha \text{ and } r_1 = 1 \text{ and } t = 1 \\ 1 & \text{if } a_1 = \beta \text{ and } r_1 = \varepsilon \text{ and } r_k = 0 \forall 1 < k \leq t \\ 0 & \text{otherwise.} \end{cases}$$

Taking action  $\alpha$  first, we have  $\nu(r_{1:t} \parallel \alpha a_{2:t}) = 0$  for  $t > 1$  (the environment  $\nu$  ends in time step 2 given history  $\alpha$ ). Hence we use (3) to conclude

$$V_{\nu}^*(\alpha) = \frac{1}{\Gamma_t} \lim_{m \rightarrow \infty} \sum_{r_{1:m}} \nu(r_{1:m} \parallel \alpha a_{2:m}) \sum_{i=1}^m r_i = 0.$$

Taking action  $\beta$  first we get

$$V_{\nu}^*(\beta) = \frac{1}{\Gamma_t} \lim_{m \rightarrow \infty} \sum_{r_{1:m}} \nu(r_{1:m} \parallel \beta a_{2:m}) \sum_{i=1}^m r_i = \frac{\gamma_1}{\Gamma_1} \varepsilon.$$

Since  $\gamma_1 > 0$  and  $\varepsilon > 0$ , we have  $V_{\nu}^*(\beta) > V_{\nu}^*(\alpha)$ , and thus AINU will use a policy that plays action  $\beta$  first, receiving a total discounted reward of  $\varepsilon\gamma_1$ . In contrast, any policy  $\pi$  that takes action  $\alpha$  first receives a larger total discounted reward of  $\gamma_1$ .  $\square$

Whether it is reasonable to assume that our environment has a nonzero probability of ending is a philosophical debate we do not want to engage in here. Instead, we have a different motivation to use the recursive value function: we get an improved computability result. Concretely, we show that for all environments  $\nu \in \mathcal{M}$ , there is a limit-computable  $\varepsilon$ -optimal policy maximizing  $\nu$ -expected rewards using an infinite horizon. According to Theorem 18, this does not hold for all  $V_{\nu}^*$ -maximizing agents AINU.

In order to maximize  $\nu$ -expected rewards in case  $\nu$  is not a measure, we need the recursive definition of the value function (analogously to [Hut05, Eq. 4.12]). To avoid confusion, we denote it  $W_{\nu}^{\pi}$ :

$$\begin{aligned} W_{\nu}^{\pi}(\mathbf{x}_{<t}) &= \frac{1}{\Gamma_t} \sum_{e_t} (\gamma_t r_t \\ &\quad + \Gamma_{t+1} W_{\nu}^{\pi}(\mathbf{x}_{1:t})) \nu(e_t \mid e_{<t} \parallel a_{1:t}) \end{aligned}$$

where  $a_t := \pi(\mathbf{x}_{<t})$ . In the following we write it in non-recursive form.

**Definition 20** ( $\nu$ -Expected Value Function). The  $\nu$ -expected value of a policy  $\pi$  in an environment  $\nu$  given

history  $\mathbf{x}_{<t}$  is

$$W_\nu^\pi(\mathbf{x}_{<t}) := \frac{1}{\Gamma_t} \sum_{m=t}^{\infty} \sum_{e_{t:m}} \gamma_m r_m \nu(e_{1:m} \parallel a_{1:m})$$

if  $\Gamma_t > 0$  and  $W_\nu^\pi(\mathbf{x}_{<t}) := 0$  if  $\Gamma_t = 0$  where  $a_i := \pi(e_{<i})$  for all  $i \geq t$ . The *optimal  $\nu$ -expected value* is defined as  $W_\nu^*(h) := \sup_\pi W_\nu^\pi(h)$ .

The difference between  $V_\nu^\pi$  and  $W_\nu^\pi$  is that for  $W_\nu^\pi$  all obtained rewards matter, but for  $V_\nu^\pi$  only the rewards in timelines that continue indefinitely. In this sense the value function  $V_\nu^\pi$  conditions on surviving forever. If the environment  $\mu$  is a measure, then the history is infinite with probability one, and so  $V_\nu^\pi$  and  $W_\nu^\pi$  coincide. Hence this distinction is not relevant for AIMU, only for AINU and AIXI.

So why use  $V_\nu^\pi$  in the first place? Historically, this is how infinite-horizon AIXI has been defined [Hut05, Def. 5.30]. This definition is the natural adaptation of (optimal) minimax search in zero-sum games to the (optimal) expectimax algorithm for stochastic environments. It turns out to be problematic only because semimeasures have positive probability of ending prematurely.

**Lemma 21** (Complexity of  $W_\nu^*$ ). *For every LSCCCS  $\nu \in \mathcal{M}$ , and every lower semicomputable discount function  $\gamma$ , the function  $W_\nu^*$  is  $\Delta_2^0$ -computable.*

*Proof.* The proof is analogous to the proof of Lemma 7. We expand Definition 20 using the expectimax operator analogously to (3). This gives a quotient with numerator

$$\lim_{m \rightarrow \infty} \max_{\mathbf{x}_{t:m}} \sum_{i=t}^m \gamma_i r_i \nu(e_{1:i} \parallel a_{1:i}),$$

and denominator  $\nu(e_{<t} \parallel a_{<t}) \cdot \Gamma_t$ . In contrast to the iterative value function, the numerator is now nondecreasing in  $m$  because we assumed rewards to be nonnegative (Assumption 3b). Hence both numerator and denominator are lower semicomputable functions, so Lemma 2 (iv) implies that  $W_\nu^*$  is  $\Delta_2^0$ -computable.  $\square$

Now we can apply our results from Section 3.2 to show that using the recursive value function  $W_\nu^\pi$ , we get a universal AI model with an infinite horizon whose  $\varepsilon$ -approximation is limit computable. Moreover, in contrast to iterative AINU, recursive AINU actually maximizes  $\nu$ -expected rewards.

**Corollary 22** (Complexity of Recursive AINU/AIXI). *For any environment  $\nu \in \mathcal{M}$ , recursive AINU is  $\Delta_3^0$  and there is an  $\varepsilon$ -optimal recursive AINU that is  $\Delta_2^0$ . In particular, for any universal Turing machine, recursive AIXI is  $\Delta_3^0$  and there is an  $\varepsilon$ -optimal recursive AIXI that is limit computable.*

*Proof.* From Theorem 8, Theorem 11, and Lemma 21.  $\square$

Analogously to Theorem 14, Theorem 16, and Theorem 17 we can show that recursive AIXI is not computable, recursive AINU is  $\Pi_2^0$ -hard, and for some universal Turing machines,  $\varepsilon$ -optimal recursive AIXI is  $\Sigma_1^0$ -hard.

## 5 DISCUSSION

We set out with the goal of finding a limit-computable perfect agent. Table 3 on page 4 summarizes our computability results regarding Solomonoff's prior  $M$ : conditional  $M$  and  $M_{\text{norm}}$  are limit computable, while  $\overline{M}$  and  $\overline{M}_{\text{norm}}$  are not. Table 1 on page 2 summarizes our computability results for AINU, AIXI, and AINU: iterative AINU with finite lifetime is  $\Delta_3^0$ . Having an infinite horizon increases the level by one, while restricting to  $\varepsilon$ -optimal policies decreases the level by one. All versions of AINU are situated between  $\Delta_2^0$  and  $\Delta_4^0$  (Corollary 9 and Corollary 12). For environments that almost surely continue forever (semimeasure that are measures), AIMU is limit-computable and  $\varepsilon$ -optimal AIMU is computable. We proved that these computability bounds on iterative AINU are generally unimprovable (Theorem 15 and Theorem 16). Additionally, we proved weaker lower bounds for AIXI independent of the universal Turing machine (Theorem 14) and for  $\varepsilon$ -optimal AIXI for specific choices of the universal Turing machine (Theorem 17 and Theorem 18).

We considered  $\varepsilon$ -optimality in order to avoid having to break argmax ties. This  $\varepsilon$  does not have to be constant over time, instead we may let  $\varepsilon \rightarrow 0$  as  $t \rightarrow \infty$  at any computable rate. With this we retain the computability results of  $\varepsilon$ -optimal policies and get that the value of the  $\varepsilon(t)$ -optimal policy  $\pi_\nu^{\varepsilon(t)}$  converges rapidly to the  $\nu$ -optimal value:  $V_\nu^*(\mathbf{x}_{<t}) - V_\nu^{\pi_\nu^{\varepsilon(t)}}(\mathbf{x}_{<t}) \rightarrow 0$  as  $t \rightarrow \infty$ . Therefore the limitation to  $\varepsilon$ -optimal policies is not very restrictive.

When the environment  $\nu$  has nonzero probability of not producing a new percept, the iterative definition (Definition 4) of AINU fails to maximize  $\nu$ -expected rewards (Proposition 19). We introduced a recursive definition of the value function for infinite horizons (Definition 20), which correctly returns  $\nu$ -expected value. The difference between the iterative value function  $V$  and recursive value function  $W$  is readily exposed in the difference between  $M$  and  $\overline{M}$ . Just like  $V$  conditions on surviving forever, so does  $\overline{M}$  eliminate the weight of programs that do not produce infinite strings. Both  $\overline{M}$  and  $V$  are not limit computable for this reason.

Our main motivation for the introduction of the recursive value function  $W$  is the improvement of the computability of optimal policies. Recursive AINU is  $\Delta_3^0$  and admits a limit-computable  $\varepsilon$ -optimal policy (Corollary 22). In this sense our goal to find a limit-computable perfect agent has been accomplished.

## REFERENCES

- [BD62] David Blackwell and Lester Dubins. Merging of opinions with increasing information. *The Annals of Mathematical Statistics*, pages 882–886, 1962.
- [Gá83] Péter Gács. On the relation between descriptive complexity and algorithmic probability. *Theoretical Computer Science*, 22(1–2):71–93, 1983.
- [Hut00] Marcus Hutter. A theory of universal artificial intelligence based on algorithmic complexity. Technical Report cs.AI/0004001, 2000. <http://arxiv.org/abs/cs.AI/0004001>.
- [Hut01] Marcus Hutter. New error bounds for Solomonoff prediction. *Journal of Computer and System Sciences*, 62(4):653–667, 2001.
- [Hut05] Marcus Hutter. *Universal Artificial Intelligence: Sequential Decisions Based on Algorithmic Probability*. Springer, 2005.
- [LH14] Tor Lattimore and Marcus Hutter. General time consistent discounting. *Theoretical Computer Science*, 519:140–154, 2014.
- [LH15a] Jan Leike and Marcus Hutter. Bad universal priors and notions of optimality. In *Conference on Learning Theory*, 2015.
- [LH15b] Jan Leike and Marcus Hutter. On the computability of Solomonoff induction and knowledge-seeking. 2015. Forthcoming.
- [LV08] Ming Li and Paul M. B. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications*. Texts in Computer Science. Springer, 3rd edition, 2008.
- [MGLA00] Martin Mundhenk, Judy Goldsmith, Christopher Lusena, and Eric Allender. Complexity of finite-horizon Markov decision process problems. *Journal of the ACM*, 47(4):681–720, 2000.
- [MHC99] Omid Madani, Steve Hanks, and Anne Condon. On the undecidability of probabilistic planning and infinite-horizon partially observable Markov decision problems. In *AAAI/IAAI*, pages 541–548, 1999.
- [MHC03] Omid Madani, Steve Hanks, and Anne Condon. On the undecidability of probabilistic planning and related stochastic optimization problems. *Artificial Intelligence*, 147(1):5–34, 2003.
- [Nie09] André Nies. *Computability and Randomness*. Oxford University Press, 2009.
- [PT87] Christos H Papadimitriou and John N Tsitsiklis. The complexity of Markov decision processes. *Mathematics of Operations Research*, 12(3):441–450, 1987.
- [RH11] Samuel Rathmanner and Marcus Hutter. A philosophical treatise of universal induction. *Entropy*, 13(6):1076–1136, 2011.
- [SB98] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.
- [SLR07] Régis Sabbadin, Jérôme Lang, and Nasolo Ravoanjanahry. Purely epistemic Markov decision processes. In *AAAI*, volume 22, pages 1057–1062, 2007.
- [Sol64] Ray Solomonoff. A formal theory of inductive inference. Parts 1 and 2. *Information and Control*, 7(1):1–22 and 224–254, 1964.
- [Sol78] Ray Solomonoff. Complexity-based induction systems: Comparisons and convergence theorems. *IEEE Transactions on Information Theory*, 24(4):422–432, 1978.
- [VNH<sup>+</sup>11] Joel Veness, Kee Siong Ng, Marcus Hutter, William Uther, and David Silver. A Monte-Carlo AIXI approximation. *Journal of Artificial Intelligence Research*, 40(1):95–142, 2011.
- [WSH11] Ian Wood, Peter Sunehag, and Marcus Hutter. (Non-)equivalence of universal priors. In *Solomonoff 85th Memorial Conference*, pages 417–425. Springer, 2011.

---

# Tracking with ranked signals

---

Tianyang Li<sup>†</sup>

Harsh Pareek<sup>†</sup>

Pradeep Ravikumar<sup>†</sup>

Dhruv Balwada<sup>\*</sup>

Kevin Speer<sup>\*</sup>

<sup>†</sup> Department of Computer Science, The University of Texas at Austin  
{lty, harshp, pradeep}@cs.utexas.edu

<sup>\*</sup> Geophysical Fluid Dynamics Institute at Florida State University  
{db10d, kspeer}@fsu.edu

## Abstract

We present a novel graphical model approach for a problem not previously considered in the machine learning literature: that of tracking with ranked signals. The problem consists of tracking a single target given observations about the target that consist of ranked continuous signals, from unlabeled sources in a cluttered environment. We introduce appropriate factors to handle the imposed ordering assumption, and also incorporate various systematic errors that can arise in this problem, particularly clutter or noise signals as well as missing signals. We show that inference in the obtained graphical model can be simplified by adding bipartite structures with appropriate factors. We apply a hybrid approach consisting of belief propagation and particle filtering in this mixed graphical model for inference and validate the approach on simulated data. We were motivated to formalize and study this problem by a key task in Oceanography, that of tracking the motion of RAFOS ocean floats, using range measurements sent from a set of fixed beacons, but where the identities of the beacons corresponding to the measurements are not known. However, unlike the usual tracking problem in artificial intelligence, there is an implicit ranking assumption among signal arrival times. Our experiments show that the proposed graphical model approach allows us to effectively leverage the problem constraints and improve tracking accuracy over baseline tracking methods yielding results similar to the ground truth hand-labeled data.

## 1 INTRODUCTION

We consider the problem of tracking a single target where the observations, concerning the position of the target, consist of ranked continuous signals from unlabeled sources in

a cluttered environment. We allow for various types of error that may occur in these observations: (a) a recorded signal may correspond to a spurious signal instead of a true signal, and (b) a signal may be lost and never recorded. We present a novel graphical model approach for this problem. Our graphical model is a mixed or hybrid model with both discrete and continuous components. To perform probabilistic inference in this mixed graphical model, we use particle filtering for the continuous component which represents the location of the target, and belief propagation for the discrete component which represents the data association between the signals and the signals' sources.

We were motivated to study the above problem formalism by a mathematical abstraction of a key problem in Oceanography: that of tracking RAFOS floats using ranked range measurements. RAFOS floats (Rossby *et al.*, 1986; Hancock & Speer, 2013) are low cost acoustically tracked subsurface floating devices used to study ocean currents by measuring the paths taken by fluid parcels in the ocean. They also measure temperature and pressure along the way. The typical mission times for these floats are on the order of a few months to a few of years during which they don't surface, and hence it is not possible to **locate or track** these floats via satellite (GPS) position fixes. In order to solve this location or tracking problem, a moored (fixed) array of sound sources, also known as *beacons* are used. These moored sound sources or beacons produce and transmit one sound signal per tracking cycle. The floats then record the **arrival times** of these signals transmitted from the beacons. These arrival times depend on the distance from the sound source and the velocity of sound in the ocean, and thus provide information about the location of the float (since the sound sources or beacons are fixed and their locations are known). At any given instant, if distances of the float from three beacons are known, this suffices to determine the position of the float. Thus if we could identify the beacons corresponding to the received signals, we can then track the course of the float.

The caveat is that while the arrival times are stored, the beacons from which the respective signals originated is **not**

known. In addition, the storage capacity of each float is limited and it only records a small number of these signal arrival times each day. These signals have a natural ordering in that the beacon signals most likely to be stored originate from the closest beacons and are received (and stored) in distance order. We are interested in inferring the identity of the beacons at each time step corresponding to the few received ordered or “ranked” signals. This is a challenging task and currently, this information is hand-labeled by oceanography researchers, with many months of effort. Our paper provides an automated solution for this interesting problem, and moreover provides a novel machine learning problem abstraction of tracking with ranked signals, which would be of interest even from a purely machine learning standpoint.

We will nonetheless anchor our discussion of the tracking with ranked signals problem to the RAFOS float tracking problem for presentational reasons. Let us consider the setup and assumptions of the above RAFOS float tracking problem in greater detail. Each float’s tracking data consists of its initial and final positions, and a fixed number of earliest signal arrival times for each day. Figure 1 shows the observed signal arrival times for a particular float over the entire tracking period. There are  $s$  fixed beacons with known positions. Each float in the ocean is equipped with a receiver. Every day at a specified time, a float starts listening for sound signals transmitted by the beacons. The float stores the arrival times of the first  $r (< s)$  signals it receives and a confidence value for each signal, then shuts off its receiver. Our goal is to use these arrival times to track the position of the float over time. The model we present includes the following kinds of errors that capture key characteristics of this problem:

- The arrival times are subject to noise due to environmental factors and recording equipment
- Signals from a beacon may never reach or be dropped by the receiver, we call these errors *missing values*.
- The receiver may erroneously record ambient noise as a signal from a beacon, and store that value. We call these errors *junk values* or *clutter*. Figure 1 and our analysis in the experiments section show that junk values are very common and outnumber true signals in the data.

While our focus is on the beacon association problem, it should be noted that there are additional sources of error such as those due to the Doppler effect, variations in the speed of sound due to temperature and beacon depth or clock drift in the receiver which have been investigated in prior work (Wooding *et al.*, 2005; Hancock & Speer, 2013). These would change the conditional probability distributions in the model we introduce in Figure 2 and incorporating these in our model is left as future work.

**Contributions** The main contributions of this work are as follows:

- Our setting differs from prior work from a machine learning standpoint due to the ordering condition imposed on the received signals, as described earlier. Our model introduces novel ranking based factors to enforce this condition, and we present an equivalent bipartite model similar in structure to that used in the data association literature (Williams & Lau, 2010) which can then be used for inference via message passing. This contribution should be valuable for other graphical model problems where ranking based factors are involved.
- RAFOS float tracking has not previously been considered from a graphical model perspective; indeed, (Hancock & Speer, 2013; Wooding *et al.*, 2005) label data by hand. Our model in Figure 2 captures several important characteristics of this problem and is also of practical importance from an application standpoint.
- We evaluate our algorithm on simulated data and show that our algorithm performs better than a baseline model which does not take the ranking of signal arrival times into account. We also present the results of our algorithm on real world RAFOS float data and demonstrate good agreement with hand-labeled data.

## 1.1 RELATED WORK

We now review prior work on tracking, and specifically contrast our problem with the classical tracking problem. A tracking algorithm can be viewed as computing the probability distribution of a system’s state  $x_t$  given observations  $y_t$ . A common model for this is a Hidden Markov model with hidden states  $x_t$ . Inference using message passing for this model leads to the well known Kalman filtering and smoothing algorithms or the forward-backward algorithm under different distributional assumptions. Another common class of approaches to perform inference on such models are particle filtering and other MCMC-based approaches (Oh *et al.*, 2009; Chertkov *et al.*, 2010).

In the RAFOS float tracking problem, from the perspective of the float, the problem resembles a multi-target tracking or data association problem, where the beacons are the targets. In multi-target tracking, we try to track a number of moving targets over time given some noisy information about the set of observed locations at each time step, and the challenge is to link the locations over time to obtain the trajectory for each target. Classical approaches to multi-target tracking assign a probability to each possible association of targets across time steps. The inference problem  $p(x_t|y_t)$  then requires summing over all possible such associations and reduces to computing the permanent of a matrix (Oh *et al.*, 2009; Chertkov *et al.*, 2010), which is

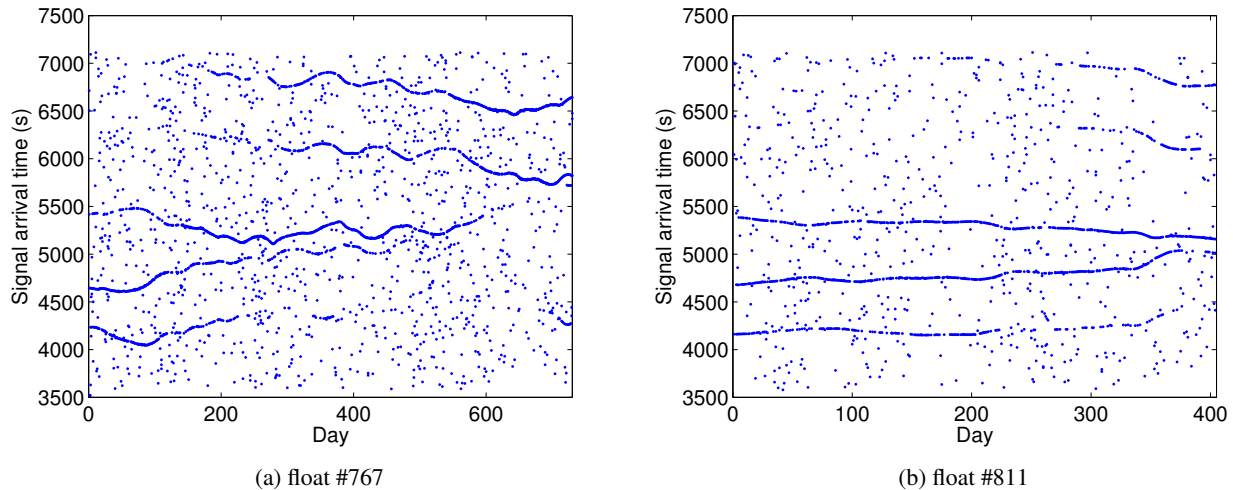


Figure 1: Observed signal arrival times for float #767 and #811 over the entire tracking period

known to be #P-complete (Valiant, 1979). The probabilistic data association filter (Bar-Shalom, 1987; Bar-Shalom *et al.*, 2009) collapses this state into a single Gaussian at each time step to make the problem tractable. Another interesting line of work is multiple hypothesis tracking (Blackman, 2004). At each time step the algorithm maintains a mixture of Gaussians for each target representing a distribution over its possible positions, and updates the state by summing over all possible associations while discarding components with low probability to make computation tractable. Message passing algorithms which have been shown to converge (Vontobel, 2013; Huang & Jebara, 2009) have also been proposed. (Cevher *et al.*, 2006) developed a particle filtering approach which uses only range measurements, while MCMC algorithms that are fully polynomial-time randomized approximation schemes (Jerum *et al.*, 2004) have also been studied. Other approaches include greedy approaches widely used in robotics, which include choosing the data association with the maximum likelihood (Thrun *et al.*, 2005), and nearest-neighbor methods where observations “closest” to expected observations are kept and others are discarded. While simple, such greedy approaches work poorly under relatively high noise levels (Bar-Shalom *et al.*, 2009). Yet another interesting line of work represents the state  $x_t$  of the system using distributions over permutations and uses group-theoretic methods to approximate these distributions (Kondor *et al.*, 2007; Huang *et al.*, 2009). Such approaches have been investigated in the contexts of radar tracking, computer vision, and robotics.

We cannot directly apply these methods to the RAFOS float tracking problem because the number of observed targets (beacon arrival times) is small compared to the number of beacons and thus most beacons are unobserved at each time step. This is because if we can identify the beacons corre-

sponding to each arrival time, only a small number (three in 2D space) of beacons are required to track the float’s position, and thus the float need only store the first few arrival times while the number of beacons may be much larger. Further, there may be regime changes, where a beacon goes out of range and a previously unobserved beacon appears in range which these methods cannot directly handle. However, since the closest beacons are the ones most likely to be recorded and their values are received in order, the provided information may be sufficient for tracking. We show in this paper that explicitly modeling the ordering assumption among beacon times and the fact that there is an underlying latent variable – the float’s position – which connects the targets, allows us to track the float.

We also note that multiple hypothesis tracking approaches which maintain a mixture over the potential associations are not required for our model for tracking with ranked signals. In a sense, in the RAFOS float tracking problem the only true latent variable is the float’s position and the others such as the beacon arrival times are derived from this. In particular, the additional ordering assumption provides a strong helpful constraint, since the only way the closest beacon’s signal is not received first is if two signals were close together and their ordering was changed due to ambient noise, if that signal is lost or if a junk signal was recorded as the first. The first possibility does not affect tracking since the two signals were already close, while the probabilities of the latter two systematic errors can be modeled in a principled manner with a graphical model approach. Thus, in the RAFOS tracking problem knowing the initial position allows us to continue tracking the float without needing a mixture model and in this sense, our model for tracking with ranked signals is (computationally and statistically) easier than the standard multiple-target tracking problem.

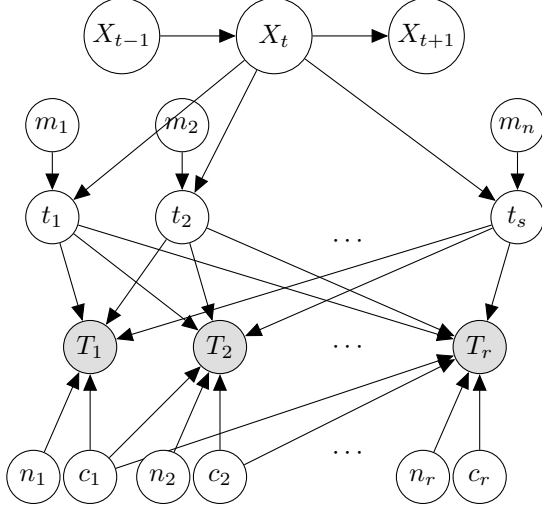


Figure 2: Graphical Model for tracking with ranked signals

To investigate the question of the practical advantage of our ordering constraint, which is a key difference from previous work, we consider a baseline model by dropping the ranking assumption. Thus, any permutation is allowed as a possible signal to signal source assignment, and is given the same weight. This approach leads to a mixture over possible assignments and we compare against this baseline in our experiments.

## 2 GRAPHICAL MODEL

Our proposed graphical model for tracking with ranked signals is shown in Figure 2. While Figure 2 expresses our model as a directed graphical model (also known as a Bayesian network), we will specify the probability distribution not in terms of conditional probabilities but in terms of local factors in the corresponding factor graph (see Figure 9 in appendix), since these will be used for inference in the sequel. For simplicity of description, we assume that target is in a 2-dimensional space, however it must be noted that our model applies to the general case when the target is in a  $d$ -dimensional space. We now describe each node and factor in Figure 2.

At each time step, the target records the first  $r$  signals emitted from  $s$  signal sources. It is possible that a signal from a particular signal source may be lost and not detectable at the target for the target to record. It is also possible that the target may record a spurious signal if the spurious signal is ranked higher than a true signal.

We indicate the position of the target by  $X_t \in \mathbb{R}^2$ . We model the system's dynamics as  $X_{t+1} \sim \mathcal{N}(X_t, \Sigma)$ , so that we have the corresponding factor  $f(X_t, X_{t+1})$  given by

$$f(X_t, X_{t+1}) = (2\pi)^{-1} |\Sigma|^{-1/2} \exp\left(-\frac{1}{2} \Delta_t^T \Sigma^{-1} \Delta_t\right) \quad (1)$$

where  $\Delta_t = X_{t+1} - X_t$ .

We denote the signal characteristic (providing information about the target) from signal source  $i$  by  $t_i \in \mathbb{R}$ , ( $1 \leq i \leq s$ ). Note that in the context of RAFOS float tracking, this would correspond to the arrival time of the sound signal from beacon  $i$ . Some of these signals might be lost as noted in the introduction, so that we use Bernoulli random variables  $m_i \in \{0, 1\}$  to indicate whether or not the signal from signal source  $i$  was lost. Thus, if  $m_i = 1$ , the signal was lost and  $t_i$  is set to  $\infty$  (so that signal source  $i$  will be ignored in (3)). Otherwise,  $t_i$  follows some given distribution depending on signal source  $i$  and current the target location  $X_t$ . For example, in the RAFOS tracking problem,  $t_i$  follows a Gaussian distribution centered on the time taken for a sound signal to travel from location  $b_i$  to location  $X_t$ , given by  $\frac{\|X_t - b_i\|_2}{v_s}$ , where  $b_i$  is the beacon's location and  $v_s$  is the speed of sound. We capture this interaction between the state  $X_t$ , the estimated arrival time  $t_i$ , and the lost signal indicator  $m_i$  via the factors  $g_i(X_t, t_i, m_i)$  ( $1 \leq i \leq s$ ):

$$g_i(X_t, t_i, m_i) = \begin{cases} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(t_i - \frac{\|X_t - b_i\|_2}{v_s})^2}{2\sigma^2}} & \text{if } m_i = 0 \\ 1 & \text{if } m_i = 1 \end{cases} \quad (2)$$

Note that  $t_i \in \mathbb{R}$ , ( $1 \leq i \leq s$ ) are not directly observed, and hence latent variables in the model. Next, we consider the *observed* signals which we denote by  $T_i$  ( $1 \leq i \leq r$ ). Since these signals are stored sequentially, we have that  $T_1 < T_2 < \dots < T_r$  by our ranking assumption. But some of these signals may not correspond to signals from actual signal sources at all, and could be purely due to *clutter*, as noted in the introduction. We thus use Bernoulli random variables  $c_i \in \{0, 1\}$  to indicate whether or not the signal  $T_i$  corresponds to clutter. By the assumptions made in the previous section, if the signal from the signal source that is supposed to be recorded first is not lost, i.e.  $t_i < \infty$ , and moreover the value stored as  $T_i$  is not junk, i.e.  $c_i = 1$  then  $T_i$  *must* be the minimum of  $\{t_j\}_{j=1}^i$ . Otherwise, a junk clutter value is recorded. We assume these clutter values follow some fixed distribution, which we denote using the random variables  $n_i$ . For instance, in the RAFOS float tracking problem we assume junk clutter value are distributed *uniformly* over a specified interval.

We represent this interaction of  $T_j$  with the clutter, lost/missing, and signal variables, via the factor  $f(T_j, c_1, c_2, \dots, c_j, n_j, t_1, m_1, t_2, m_2, \dots, t_s, m_s)$ :

$$f(T_j, c_1, c_2, \dots, c_j, n_j, t_1, m_1, t_2, m_2, \dots, t_s, m_s) = \begin{cases} \delta(T_j - t_{(j)}) & \text{if } c_j = 0 \\ \delta(T_j - n_j) & \text{if } c_j = 1 \end{cases} \quad (3)$$

where we use  $t_{(j)}$  to denote the  $(j - \sum_{l=1}^{j-1} c_l)^{\text{th}}$  smallest element of the set  $\{t_l\}$ . This is a degenerate conditional

probability distribution for  $T_i$  with mass only at the appropriate  $t_i$  or  $n_i$ . Recall that  $\delta$  is the Dirac delta function, which has the following properties:

$$\delta(x) = \begin{cases} 0 & \text{if } x \neq 0 \\ \text{undefined} & \text{if } x = 0 \end{cases}, \int \delta(x)dx = 1, \\ \int f(x)\delta(x)dx = f(0) \quad (4)$$

## 2.1 INFERENCE WITH $\min$ FACTORS

The key novelty of our graphical model are the high order factors  $f_j$ , which depend on the  $r$  first ranked signals. We consider this model in the simplified case, with no missing signals and no clutter, and show how inference via message passing can be performed in this model. In this setting, the factor  $f_j(T_j, t_1, t_2, \dots, t_s)$  is given by

$$f_j(T_j, t_1, t_2, \dots, t_s) = \delta(T_j - t_{(j)}) \quad (5)$$

where  $t_{(j)}$  is the  $j^{\text{th}}$  minimum element of  $\{t_1, t_2, \dots, t_s\}$ .

Direct computation of messages for high order factors in general requires computing an  $s - 1$ -dimensional integral. However, our  $f_j$ , which correspond to the  $j$ -th minimum function, can be rewritten as a sum of products as,

$$f_j = \sum_{k=1}^s \delta(t_k - T_j) \\ \sum_{(A,B) \in \mathcal{S}_k} \prod_{a \in A} \mathbf{1}(t_a < T_j) \prod_{b \in B} \mathbf{1}(t_b > T_j) \quad (6)$$

where  $\mathcal{S}_k = \{(A, B) \subseteq [s] \times [s] : A \cup B = [s] \setminus \{k\}, A \cap B = \emptyset, |A| = j - 1, |B| = s - j\}$  and  $[s] = \{1, 2, \dots, s\}$

Then, as we show in the appendix, the multidimensional integral, comprising messages in a message passing inference algorithm, can be computed using only one-dimensional integrals and turn out to have the form:

$$\mu_{f_j \rightarrow t_i}(t_i) = \delta(t_i - T_j)h_1(T_j) + \mathbf{1}(t_i < T_j)h_2(T_j) \\ + \mathbf{1}(t_i > T_j)h_3(T_j) \quad (7)$$

where each  $h_j$  can be computed in  $O(s^r)$  time via dynamic programming. When  $r$  is fixed, as in the RAFOS float tracking problem – since with  $r = 3$  at most three known nearest beacons we can obtain the floats position and the overall number of beacons  $s$  is irrelevant – these messages can effectively be computed in polynomial time. This result is in the vein of previous work on reductions for high order potentials such as (Tarlow *et al.*, 2010). Handling high order min factors in this way is a novel result and can potentially be applied to other problems where such ranking factors are involved. Due to their technical complexity however, we defer further discussion of these factors and the details of the message passing algorithm to Appendix A.2. Instead, in the next section, we consider a reduction of our

graphical model, with many such min factors, to an auxiliary model with a bipartite graph structure, which results in easier to implement algorithms.

## 2.2 REDUCTION TO BIPARTITE FACTORS

In this section, we show that our Bayesian network can be represented via a simpler conditional random field with factors that enforce a bipartite matching constraint. We do this by introducing latent variables  $S_i, R_j$  (Figure 3) to represent the association between signal sources and signals.

**One large factor.** First, consider the simplified case with no clutter and no missing signals. In this case, our graphical model is equivalent to one with the following large factor  $f$  connecting the  $t_1, t_2, \dots, t_s$  and  $T_1, T_2, \dots, T_r$  (instead of the  $f_j$  of (3) connecting  $t_j$  to the  $\{T_i\}$ ,  $f(t_1, t_2, \dots, t_s, T_1, T_2, \dots, T_r)$  as

$$f(t_1, t_2, \dots, t_s, T_1, T_2, \dots, T_r) \\ = \sum_{\substack{\Pi \in \{\text{r-permutations} \\ \text{of } \{1, 2, \dots, s\}\}}} \left\{ \prod_{j=1}^r \delta(t_{\Pi(j)} - T_j) \prod_{k \notin \Pi} \mathbf{1}(t_k > T_r) \right\} \quad (8)$$

We prove this by showing that the factor  $f$  defined in (8) above satisfies:

$$f = \prod_{j=1}^r f_j \quad (9)$$

where  $f_1, f_2, \dots, f_j$  are the factors in eq (3). For any  $\{t_1, t_2, \dots, t_s\}$ , if the  $r$  smallest elements are not  $T_1, T_2, \dots, T_r$  then we can see that  $f = 0$  and  $\prod_{j=1}^r f_j = 0$ . When the  $r$  smallest elements of  $\{t_1, t_2, \dots, t_s\}$  are  $T_1, T_2, \dots, T_r$ , where  $t_{i_1} = T_1, t_{i_2} = T_2, \dots, t_{i_r} = T_r$ , then we can see that  $f = \prod_{j=1}^r \delta(t_{i_j} - T_j) = \prod_{j=1}^r f_j$ .

**Auxiliary Bipartite Graph.**  $f$  has a special structure in that it sums over *partial* matchings and for each matching, it can be written as a product of pairwise factors, represented by the snippet of a pairwise conditional random field shown in Figure 3. To represent the matching, we use variables  $S_i$  ( $1 \leq i \leq s$ ) corresponding to signal sources: these take values in  $\{1, 2, \dots, r, \perp\}$ , indicating which of the  $r$  received signals  $S_i$  corresponds to, or a value of  $\perp$  if the signal was *late*, i.e. not in the first  $r$  and was thus not received. On the other side of the matching, we have variables  $R_j$  ( $1 \leq j \leq r$ ) corresponding to received signals  $T_j$ : these take values in  $\{1, 2, \dots, s\}$  indicating which of the signal sources  $T_j$  originated from. Thus,  $T_j$  is the signal corresponding to source  $R_j$ , while  $T_{S_i}$  is the signal corresponding to source  $i$ .

The factors  $f(S_i, R_j)$  enforce these bipartite matching con-



straint:

$$f(S_i, R_j) = \begin{cases} 0 & \text{if } S_i = j, R_j \neq i \text{ or} \\ & S_i \neq j, R_j = i \\ 1 & \text{otherwise} \end{cases} \quad (10)$$

And the factor  $f(S_i)$  is given by

$$f(S_i) = \begin{cases} \delta(t_i - T_{S_i}) & \text{if } S_i \neq \mathbb{L} \\ \mathbf{1}(t_i > T_r) & \text{if } S_i = \mathbb{L} \end{cases} \quad (11)$$

Then, on marginalizing  $S_i$  and  $R_j$ , we obtain the factor  $f$  as in equation (8). Thus, we can replace the factor  $f$  of equation (8) by this auxiliary graphical model and perform message passing on this graphical model. This leads to a model similar to the pairwise models proposed to approximate the permanent, which shows up in the usual data association problem (Pasula *et al.*, 1999; Huang & Jebara, 2009; Oh *et al.*, 2009; Chertkov *et al.*, 2010; Vontobel, 2013). We can now consider the difference between our bipartite model and the one used to approximate the permanent (in (Huang & Jebara, 2009)): the sum in eq (8) is over partial permutations while that in (Huang & Jebara, 2009) involves full permutations. Our model consequently involves an additional value  $L$  to indicate that a signal is ‘‘late’’, leading to the node factors as shown in eq (11) which use indicator functions to indicate that a sent signal may not be in the top  $r$  ranked signals, and thus encodes the min factors required by our model.

**Generalization to Factors with Missing Values, Clutter.** We now generalize this argument to yield a similar conditional random field with pairwise factors for the complete model with missing values and clutter. Each  $S_i$  ( $1 \leq i \leq s$ ) takes values  $1, 2, \dots, r, \mathbb{L}, \mathbb{M}$ , where a number indicates which signal it corresponds to,  $\mathbb{L}$  indicates that the signal is ranked too low to be observed, and  $\mathbb{M}$  indicates that the signal is missing and undetectable at the receiver. Each  $R_j$  ( $1 \leq j \leq r$ ) takes values  $1, 2, \dots, s, \mathbb{C}$ , where a number indicates the signal’s source, and  $\mathbb{C}$  indicates it’s clutter. The factors  $f(S_i, R_j)$  still enforce the bipartite matching constraint of equation (10). The factors  $f(n_j, R_j)$  are given by

$$f(n_j, R_j) = \begin{cases} P_C \delta(n_j - T_j) & \text{if } R_j = \mathbb{C} \\ 1 - P_C & \text{otherwise} \end{cases} \quad (12)$$

where  $P_N$  is the probability that the signal is clutter.

And the factor  $f(t_i, S_i)$  is given by

$$f(t_i, S_i) = \begin{cases} 1 - P_D & \text{if } S_i = \mathbb{M} \\ P_D \mathbf{1}(t_i > T_r) & \text{if } S_i = \mathbb{L} \\ P_D \delta(t_i - T_{S_i}) & \text{otherwise} \end{cases} \quad (13)$$

where  $P_D$  is the probability that the signal can be detected

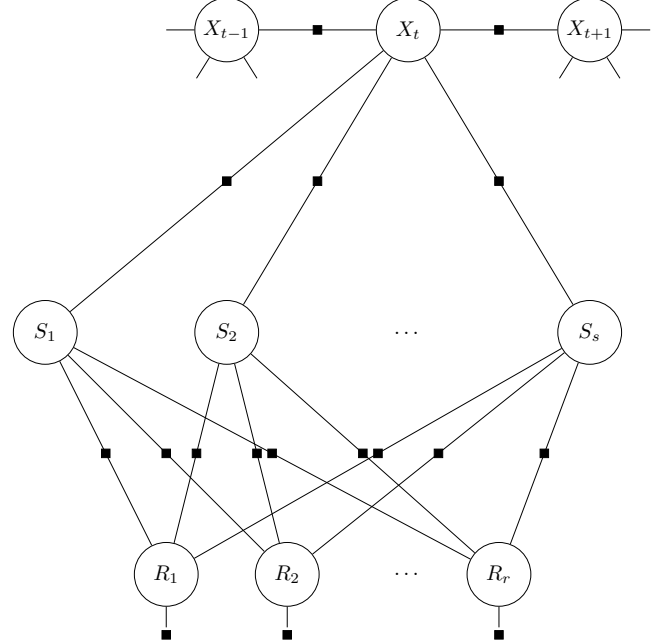


Figure 3: A conditional random field representing the distribution of  $r$  earliest arriving signals

Finally, we marginalize out  $t_1, t_2, \dots, t_s$  and  $n_1, n_2, \dots, n_r$ , so that we can represent the conditional distribution as a simpler pairwise conditional random field as in Figure 3, where no factor uses the delta function, simplifying computation.

In the conditional random field shown in Figure 3, the factor  $f(X_t, S_i)$  is given by

$$f(X_t, S_i) = \begin{cases} 1 - P_D & \text{if } S_i = \mathbb{M} \\ P_D \int_{T_r}^{+\infty} p_i(T|X_t) dT & \text{if } S_i = \mathbb{L} \\ P_D p_i(T_{S_i}|X_t) & \text{otherwise} \end{cases} \quad (14)$$

In the RAFOS tracking problem  $p_i(T|X_t) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left\{-\frac{(T - \frac{\|X_t - b_i\|_2}{v_s})^2}{2\sigma^2}\right\}$  is the distribution of signal arrival time  $T$  given the location  $X_t$  of the target, and the factor  $f(X_t, R_j)$  is given by

$$f(R_j) = \begin{cases} P_C p_N(T_j) & \text{if } R_j = \mathbb{C} \\ 1 - P_C & \text{otherwise} \end{cases} \quad (15)$$

where  $p_N(T)$  is the distribution of clutter signal arrival time.

**Alternative Derivation of the Bipartite Model** Here we show an alternative and more intuitive derivation of the bipartite model assuming that the number of clutter signals follows a Poisson distribution with parameter  $\lambda$  and clutter signals are independent and identically distributed. We also assume that the distribution of signals from different

signal sources are independent, and whether or not a signal is missing and non-detectable at the target is independent of all other signals.

Assume that the target's location is  $X$ , and there are  $N$  clutter signals, the likelihood of observing  $T_1 \leq T_2 \leq \dots \leq T_r$  with labels  $S_1, S_r, \dots, S_s$  and  $R_1, R_2, \dots, R_r$  is proportional to

$$\ell_N = \frac{N!}{(N - \sum \mathbf{1}(R_j = C))!} (1 - P_D)^{\sum \mathbf{1}(S_i = M)} P_D^{\sum \mathbf{1}(S_i \neq M)} \left( \int_{T_r}^{+\infty} p_N(t) dt \right)^{N - \sum \mathbf{1}(R_j = C)} \prod_{R_j = C} p_N(T_j) \prod_{S_i \neq L, M} p_i(T_{S_i} | X) \prod_{S_i = L} \int_{T_r}^{+\infty} p_i(t | X) dt \quad (16)$$

Marginalizing out  $N$  we then have

$$\sum_N e^{-\lambda} \frac{\lambda^N}{N!} \ell_N \propto \lambda^{\sum \mathbf{1}(R_j = C)} (1 - P_D)^{\sum \mathbf{1}(S_i = M)} P_D^{\sum \mathbf{1}(S_i \neq M)} \prod_{R_j = C} p_N(T_j) \prod_{S_i \neq L, M} p_i(T_{S_i} | X) \prod_{S_i = L} \int_{T_r}^{+\infty} p_i(t | X) dt \quad (17)$$

Thus if we set

$$P_C = \frac{\lambda}{1 + \lambda} \quad (18)$$

then it is clear that we can represent this distribution using the conditional random field shown in Figure 3.

### 2.3 INFERENCE

We use a hybrid particle filtering and belief propagation approach. For propagating information across time steps, i.e. between the nodes  $X_{t-1}$  and  $X_t$ , we use particle filtering (Doucet & Johansen, 2009). Particle based methods are simple to implement for tracking with range measurements (Cevher *et al.*, 2006). Such methods have also been widely used in robotics (Thrun *et al.*, 2005).

The distribution for each  $X_t$  is represented by a set of particles, we use message passing in the graphical model shown in Figure 3 to compute  $P(T_1, T_2, \dots, T_r | X_t)$ , which is given by the partition function and can be approximated by the Bethe free energy (Huang & Jebara, 2009). In this graphical model, message passing equations have a simple form where each iteration is  $O(sr)$  (Huang & Jebara, 2009; Williams & Lau, 2010), and message passing converges to a unique fixed point (Huang & Jebara, 2009; Williams & Lau, 2010; Vontobel, 2013). Previous work (Huang & Jebara, 2009; Williams & Lau, 2010; Chertkov *et al.*, 2010) has shown this strategy to be very effective.

---

### Algorithm 1 Algorithm for tracking with ranked signals

---

```

1: function TRACKWITHRANKEDSIGNALS( $X_0$ ,
   { $T_1^{(i)} \leq T_2^{(i)} \leq \dots \leq T_r^{(i)}\}_{i=1}^n$ )
2:    $N_p \leftarrow$  number of particles to use
3:    $p[N_p] \leftarrow$  each particle is initialized to  $X_0$ 
4:    $w[N_p] \leftarrow$  each particle's weight
5:   for  $i = 1; i \leq n; ++i$  do
6:     for  $j = 0; j \neq N_p; ++j$  do
7:        $p[j] \leftarrow$  SAMPLE( $P(X_t | X_{t-1} = p[j])$ )
8:        $w[j] \leftarrow P(T_1^{(i)}, T_2^{(i)}, \dots, T_r^{(i)} | p[j])$   $\triangleright$ 
         computed using belief propagation in the conditional
         random field in Figure 3
9:     end for
10:     $p \leftarrow$  RESAMPLE( $p, w$ )  $\triangleright$  particle filter
        resampling
11:    end for
12: end function

```

---

An implementation of the tracking algorithm to estimate  $X_1, X_2, \dots, X_n$  where we are given the initial position of the target  $X_0$ , and  $n$  observations of ranked signals  $\{T_1^{(i)} \leq T_2^{(i)} \leq \dots \leq T_r^{(i)}\}_{i=1}^n$  is given in Algorithm 1.

## 3 EXPERIMENTS

We present experiments on two kinds of datasets: simulated data generated using our model (Figure 3) and real world data from RAFOS floats.

### 3.1 SIMULATIONS

We simulate tracking RAFOS floats using ranked continuous range measurements with our method. At each time step, the target records the arrival times of the first 4 signals emitted from 10 fixed beacons. There is a fixed probability that a signal may not be detectable at the target, and the target may record the arrival time of a spurious signal (clutter) if the spurious signal arrives before an actual signal.

#### 3.1.1 SIMULATION OF TRACKING DIFFERENT TRAJECTORIES

Figure 4a shows our algorithm tracking a target moving in a straight line: the  $x$  and  $y$  axes correspond to the  $x$  and  $y$  coordinates of the 2D location of the signal. The corresponding signal arrival times are shown in Figure 4b. Similarly, Figure 5a shows our algorithm tracking a target moving in a spiral, signal arrival times are shown in Figure 5b. In the simulations, there are 10 signal sources and 4 signal arrival times. Clutter arrival times are uniformly distributed on a predefined interval, and the number of clutter signals follows a Poisson distribution with parameter 4, so that  $P_C = \frac{4}{1+4} = 0.8$ . The signal arrival time distribution  $p_i$  has parameter  $\sigma = 0.02$ , and  $P_D = 0.7$ . We can see that our algorithm correctly tracks the target when the

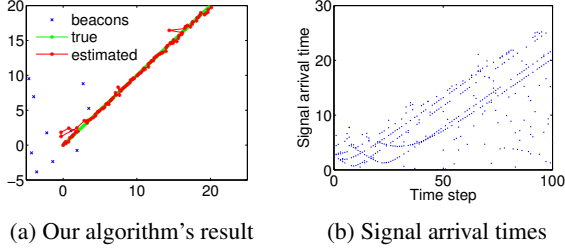


Figure 4: Our algorithm tracking a target moving in a straight line

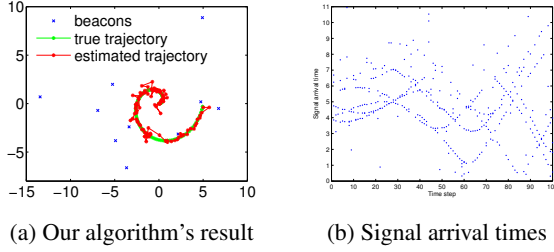


Figure 5: Our algorithm tracking a target moving in a spiral

association between signal arrival times and signal sources changes.

### 3.1.2 COMPARISON WITH THE BASELINE ALGORITHM

We compare our algorithm against a baseline algorithm which does not take into account that signal arrival times are ranked. The overall structure of the baseline model is the same as that of the graphical model shown in Figure 3, however the factor  $f(X_t, S_i)$  is given by

$$f(X_t, S_i) = \begin{cases} P_D p_i(T_{S_i} | X_t) & \text{if } S_i \neq \text{L and } S_i \neq \text{M} \\ 1 - P_D & \text{if } S_i = \text{L or } S_i = \text{M} \end{cases} \quad (19)$$

so that the implicit ranking of signal arrival times is not taken into account.

Figure 6b shows the mean squared error against the noise level when tracking the straight trajectory shown in Figure 6a. The noise level is indicated by a parameter  $\lambda$  such that  $P_C = \frac{\lambda}{1+\lambda}$ . The parameter  $\lambda$  corresponds to the Poisson parameter for the number of clutter signals (see Appendix). Signal arrival time distribution  $p_i$  has parameter  $\sigma = 0.02$ , and  $P_D = 0.7$ . We can see that our algorithm outperforms the baseline algorithm.

## 3.2 RAFOS FLOAT DATA

We present tracking results on real world RAFOS float data which consists of ranked continuous signal arrival times. We compare the results of our algorithm against hand labeled data where each signal is hand labeled with a par-

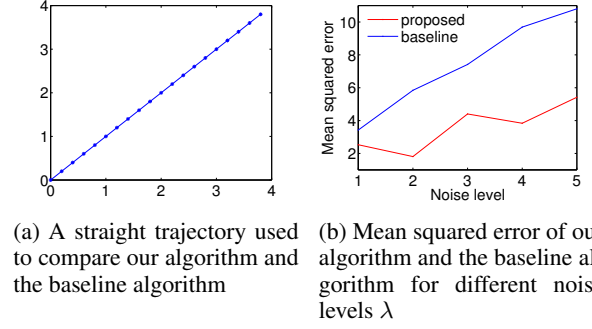


Figure 6: Simulation comparing our proposed algorithm and the baseline

ticular signal source or as clutter. In RAFOS float missions, after a float is released at a known location to start its mission, it records a fixed number of arrival times from a set of beacons whose positions are fixed and known. The dataset presented here are collected in the DIMES (Diapycnal and Isopycnal Mixing Experiment in the Southern Ocean) project (Hancock & Speer, 2013), which is aimed at measuring diapycnal and isopycnal mixing in the Southern Ocean, along the tilting isopycnals of the Antarctic Circumpolar Current. In our dataset, the float records 4 earliest signal arrival times, and there are 10 beacons. The floats are first released off the western coast of South America. There were just two floats — #767 and #811 — for which at least three beacon signals were recorded at all time steps (note that at least three beacon signals are required to uniquely identify the 2D position of the float), and accordingly, we present our results on tracking these two floats. Their signal arrival times are shown in Figure 1. We compare our algorithm's results to a manual tracking procedure where each signal is hand labeled to its source or as noise based on intuition and some apriori knowledge of the physical factors (geometry of ocean basin and acoustic array, basic knowledge of current directions etc.) at play.

Note that the *true* locations of the RAFOS floats are not known and that the hand labeled data only contains the associations between signal arrival times and their corresponding signal sources (or whether they are clutter). Hence, we obtain a trajectory for the hand labeled data using a simple particle filter. At each time step  $T_1 \leq T_2, \dots \leq T_r$  and  $R_1, R_2, \dots, R_r$  are known, so the weight of a particle at  $X$  is given by

$$w(X) \propto \prod_{1 \leq j \leq r} p_{R_j}(T_j | X) \quad (20)$$

When running our proposed algorithm and the one using hand labeled data for RAFOS float tracking, we set  $P_C = 0.5$ ,  $P_D = 0.9$ ,  $\sigma = 5$ . Although in the real environment the speed of sound depends on the depth of the RAFOS float and other environmental factors, in our experiments we set  $v_s = 1.5$  km/s across all cases.

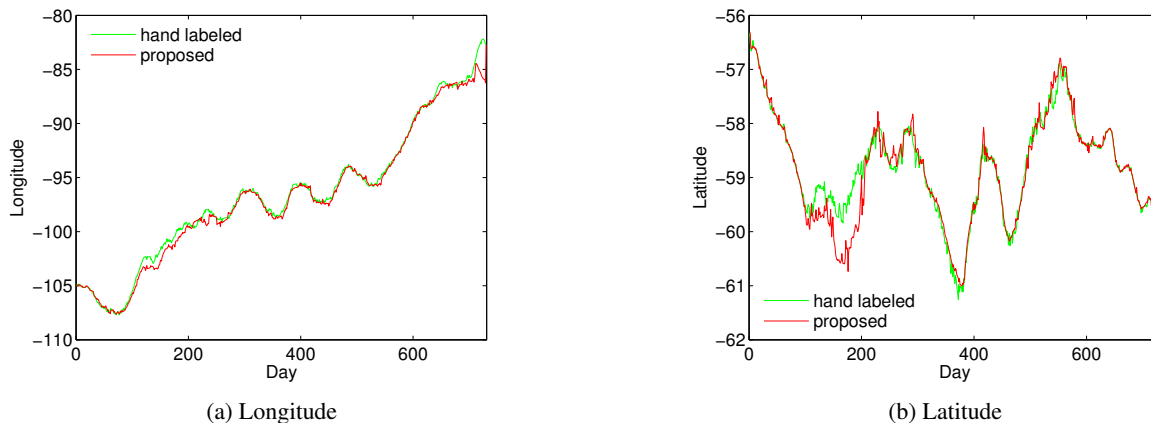


Figure 7: Trajectory of float #767 estimated by our proposed algorithm versus using hand labeled data

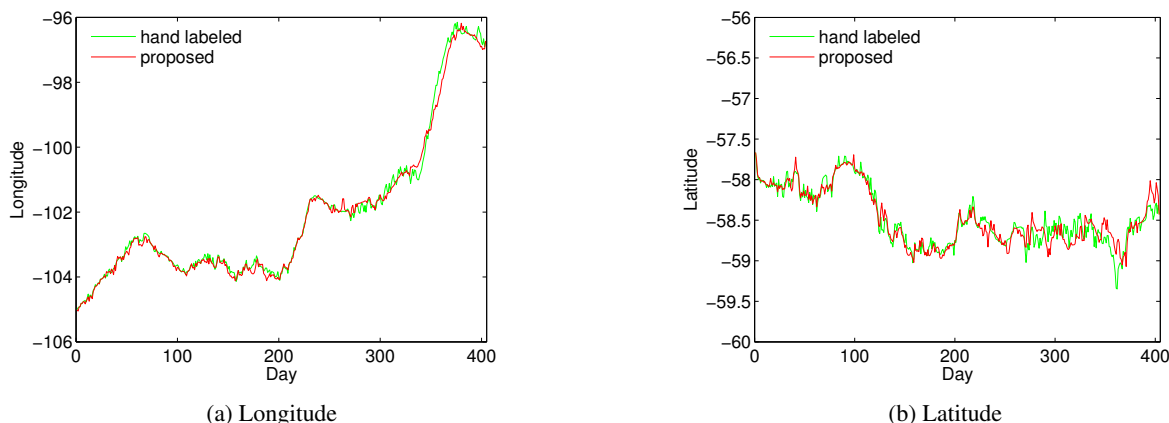


Figure 8: Trajectory of float #811 estimated by our proposed algorithm versus using hand labeled data

The results for longitude and latitude of float #767 and #811 estimated by our algorithm versus using hand labeled data are shown in Figure 7 and Figure 8 respectively. Additional results for other floats in the DIMES project are presented in the appendix. We observe good agreement between the tracks estimated using hand labeled data and using our method indicating that our method recovers the associations. We can also see that our algorithm continues to track the target when the set of associated signal sources changes.

#### 4 CONCLUSIONS

We have presented a novel graphical model approach for the problem of tracking with ranked signals which was able to capture certain problem specific features easily. The key novelty in the model, from a machine learning point of view, was the presence of ranking-based factors. We have provided a novel bipartite construction which allows for easy inference via message passing for such factors. While our model and approach were motivated by a particular application, the contributions of this paper should

be applicable to other problems where ranking based high-order factors are involved.

We experimentally showed that our method effectively leverages the varied problem constraints to improve tracking accuracy over baseline tracking methods. We applied our method for tracking with ranked signals to a key Oceanography problem of tracking RAFOS floats in the ocean and thus provide an automated solution to a problem which has previously required significant manual effort.

#### ACKNOWLEDGEMENTS

T.L., H.P., and P.R. acknowledge the support of ARO via W911NF-12-1-0390 and NSF via IIS-1149803, IIS-1320894, IIS-1447574, and DMS-1264033. K.S. and D.B. acknowledge support from NSF OCE 1231803.

## References

- Bar-Shalom, Y., Daum, F., & Huang, J. 2009. The probabilistic data association filter. *Control Systems, IEEE*, **29**(6), 82–100.
- Bar-Shalom, Yaakov. 1987. *Tracking and data association*. Academic Press Professional, Inc.
- Blackman, Samuel S. 2004. Multiple hypothesis tracking for multiple target tracking. *Aerospace and Electronic Systems Magazine, IEEE*, **19**(1), 5–18.
- Cevher, Volkan, Velmurugan, Rajbabu, & McClellan, James H. 2006. A range-only multiple target particle filter tracker. *Pages IV–IV of: Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, vol. 4. IEEE.
- Chertkov, M, Kroc, L, Krzakala, F, Vergassola, M, & Zdeborová, L. 2010. Inference in particle tracking experiments by passing messages between images. *Proceedings of the National Academy of Sciences*, **107**(17), 7663–7668.
- Doucet, Arnaud, & Johansen, Adam M. 2009. A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of Nonlinear Filtering*, **12**, 656–704.
- Hancock, C., & Speer, K. 2013. Diapycnal and Isopycnal Mixing Experiment in the Southern Ocean, RAFOS Float Data Report, Marine Field Group, FSU, January 2013.
- Huang, Bert, & Jebara, Tony. 2009. Approximating the permanent with belief propagation. *arXiv preprint arXiv:0908.1769*.
- Huang, Jonathan, Guestrin, Carlos, & Guibas, Leonidas. 2009. Fourier theoretic probabilistic inference over permutations. *The Journal of Machine Learning Research*, **10**, 997–1070.
- Jerrum, Mark, Sinclair, Alistair, & Vigoda, Eric. 2004. A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries. *Journal of the ACM (JACM)*, **51**(4), 671–697.
- Kondor, Risi, Howard, Andrew, & Jebara, Tony. 2007. Multi-object tracking with representations of the symmetric group. *Pages 211–218 of: International Conference on Artificial Intelligence and Statistics*.
- Oh, Songhwai, Russell, Stuart, & Sastry, Shankar. 2009. Markov chain Monte Carlo data association for multi-target tracking. *Automatic Control, IEEE Transactions on*, **54**(3), 481–497.
- Pasula, Hanna, Russell, Stuart, Ostland, Michael, & Ritov, Yaacov. 1999. Tracking many objects with many sensors. *Pages 1160–1171 of: IJCAI*, vol. 99.
- Rosby, T, Dorson, D, & Fontaine, J. 1986. The RAFOS system. *Journal of Atmospheric and Oceanic Technology*, **3**(4), 672–679.
- Tarlow, Daniel, Givoni, Inmar E, & Zemel, Richard S. 2010. Hop-map: Efficient message passing with high order potentials. *Pages 812–819 of: International Conference on Artificial Intelligence and Statistics*.
- Thrun, Sebastian, Burgard, Wolfram, & Fox, Dieter. 2005. *Probabilistic robotics*. MIT press.
- Valiant, Leslie G. 1979. The complexity of computing the permanent. *Theoretical computer science*, **8**(2), 189–201.
- Vontobel, Pascal O. 2013. The Bethe permanent of a non-negative matrix. *Information Theory, IEEE Transactions on*, **59**(3), 1866–1901.
- Williams, Jason L, & Lau, Roslyn A. 2010. Convergence of loopy belief propagation for data association. *Pages 175–180 of: Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2010 Sixth International Conference on*. IEEE.
- Wooding, Christine M, Furey, Heather H, & Pacheco, Marguerite A. 2005. *RAFOS float processing at the Woods Hole Oceanographic Institution*. Tech. rept. Woods Hole Oceanographic Institution.

---

# Classification of Sparse and Irregularly Sampled Time Series with Mixtures of Expected Gaussian Kernels and Random Features

---

Steven Cheng-Xian Li   Benjamin Marlin  
College of Information and Computer Sciences  
University of Massachusetts Amherst  
Amherst, MA 01003  
{cxl, marlin}@cs.umass.edu

## Abstract

This paper presents a kernel-based framework for classification of sparse and irregularly sampled time series. The properties of such time series can result in substantial uncertainty about the values of the underlying temporal processes, while making the data difficult to deal with using standard classification methods that assume fixed-dimensional feature spaces. To address these challenges, we propose to first re-represent each time series through the Gaussian process (GP) posterior it induces under a GP regression model. We then define kernels over the space of GP posteriors and apply standard kernel-based classification. Our primary contributions are (i) the development of a kernel between GPs based on the mixture of kernels between their finite marginals, (ii) the development and analysis of extensions of random Fourier features for scaling the proposed kernel to large-scale data, and (iii) an extensive empirical analysis of both the classification performance and scalability of our proposed approach.

## 1 INTRODUCTION

In this paper, we address the problem of classification of sparse and irregularly sampled time series. Irregularly sampled (or non-uniformly sampled) time series are characterized by variable time intervals between successive observations. While all time series in a data set are typically defined on the same continuous-time interval, the number of observations per time series can vary. When the intervals between successive observations are long, the time series are said to be sparsely sampled.

Such time series data arise when sampling complex temporal processes in a number of important areas including climate science [Schulz and Stattegger, 1997], ecology [Clark

and Bjørnstad, 2004], biology [Ruf, 1999], medicine [Marlin et al., 2012] and astronomy [Scargle, 1982]. In domains including medicine, the data are both irregularly sampled and sparsely sampled [Marlin et al., 2012]. Classification in this setting is challenging both because the data cases are not naturally defined in a fixed-dimensional feature space due to irregular sampling and variable time series length, and because there can be substantial uncertainty about the underlying temporal processes due to the sparsity of observations.

To address these challenges, we begin by re-representing each input time series using the Gaussian process (GP) posterior it induces under a GP regression model [Rasmussen and Williams, 2006]. We then define kernels over the space of GP posteriors and apply standard kernel-based classification methods [Cortes and Vapnik, 1995]. We propose a kernel between GPs based on the mixture of kernels between finite dimensional GP marginal distributions defined over sliding time windows. We refer to this as the *mixture of sliding GP marginal kernels (MSM)* framework.

The MSM kernel framework requires a base kernel between finite-dimensional GP marginals, which are Gaussian distributions. While the MSM framework can be used with any valid base kernel between two Gaussian distributions, we propose an uncertainty-aware base kernel that we refer to as the *expected Gaussian kernel* to address the uncertainty that results from sparse sampling. Using the expected Gaussian kernel in the MSM framework yields a kernel between Gaussian processes that we refer to as the *mixture of expected Gaussian kernels (MEG)*.

Next, we consider the problem of scaling our proposed kernel-based time series classification framework to large-scale data. Computing the exact Gram matrix for the proposed MEG kernel with  $n$  time series takes  $\mathcal{O}(n^2 d^3 k)$  time when  $k$  sliding windows of length  $d$  are used. To address this problem, we show how to extend the random Fourier feature kernel approximation framework [Rahimi and Recht, 2007] to the MEG kernel. Using our random feature construction, it takes  $\mathcal{O}(nMd^2)$  time to compute the random feature matrix using  $M$  random features for

each time series. We then show how to use Fastfood [Le et al., 2013] to reduce the random feature computation time to  $\mathcal{O}(nMd \log d)$  when the window size  $d$  is large. With an extra rank- $r$  covariance approximation, we can further reduce this time to  $\mathcal{O}(nMr \log d)$  for  $r \ll d$ . Finally, we provide a convergence analysis of our proposed approximation based on the recently developed matrix Bernstein inequality [Lopez-Paz et al., 2014; Tropp, 2012a].

The primary contributions of this paper are:

1. The development of an uncertainty-aware kernel for GPs based on the mixture of kernels between their finite-dimensional marginals (Section 3).
2. The development and analysis of an extension of random Fourier features for scaling the proposed kernel to large-scale data (Section 4).
3. An extensive empirical analysis of both the classification performance and scalability of our proposed approach (Section 5).

In Section 2, we begin by describing how to represent sparse and irregularly sampled time series using Gaussian processes, which are a fundamental building block of our proposed framework.

## 2 SPARSE AND IRREGULARLY SAMPLED TIME SERIES

Our focus in this paper is classification of time series data in the presence of sparse and irregular sampling. Consider a data set of  $n$  independent time series  $\mathcal{D} = \{\mathcal{S}_1, \dots, \mathcal{S}_n\}$ , where each time series  $\mathcal{S}_i$  is represented as a list of time points  $\mathbf{t}_i = [t_{i1}, \dots, t_{i|\mathcal{S}_i}|]^\top$ , and a list of corresponding values  $\mathbf{y}_i = [y_{i1}, \dots, y_{i|\mathcal{S}_i}|]^\top$ . We assume that each time series is defined over a common continuous-time interval  $[0, T]$ . However, for irregularly sampled time series we do not assume that all of the time series are defined on the same collection of time points (i.e.,  $\mathbf{t}_i \neq \mathbf{t}_j$  in general), we do not assume that the intervals between time points are uniform, and we do not assume that the number of observations in different time series is the same (i.e.,  $|\mathcal{S}_i| \neq |\mathcal{S}_j|$  in general).

Learning in this setting is challenging because the data cases are not naturally defined in a fixed-dimensional feature space due to irregular sampling, and there can be substantial uncertainty about the underlying temporal processes due to the sparsity of observations.

To address these challenges, we begin by re-representing each input time series using the Gaussian process (GP) posterior it induces under a Gaussian process regression model [Rasmussen and Williams, 2006]. This construction naturally accommodates sparse and irregularly sampled time

series. To obtain the posterior GPs, we use a Gaussian likelihood function with noise variance  $\sigma^2$ , and a zero-mean GP prior with the squared exponential covariance function

$$\mathcal{K}_{\text{SE}}(t_i, t_j) = a \exp(-b(t_i - t_j)^2), \text{ for } a, b > 0.$$

We learn the hyperparameters  $a, b, \sigma$  of the GP regression model by maximizing the marginal likelihood of the data.<sup>1</sup> Under this model, the posterior distribution induced by each time series  $\mathcal{S}$  is also a Gaussian process. By definition, any finite marginal of a GP is Gaussian distributed. Let  $\mathcal{GP}(\mathbf{u} | \mathcal{S} = \{\mathbf{t}, \mathbf{y}\}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$  denote the posterior GP marginal of  $\mathcal{S}$  over a collection of time points  $\mathbf{u}$ . The mean and covariance of the Gaussian posterior marginal  $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$  is given below where  $[K(\mathbf{u}, \mathbf{t})]_{ij} = \mathcal{K}_{\text{SE}}(u_i, t_j)$ .

$$\begin{aligned} \boldsymbol{\mu} &= K(\mathbf{u}, \mathbf{t}) (K(\mathbf{t}, \mathbf{t}) + \sigma^2 \mathbf{I})^{-1} \mathbf{y}, \\ \boldsymbol{\Sigma} &= K(\mathbf{u}, \mathbf{u}) - K(\mathbf{u}, \mathbf{t}) (K(\mathbf{t}, \mathbf{t}) + \sigma^2 \mathbf{I})^{-1} K(\mathbf{t}, \mathbf{u}). \end{aligned} \quad (1)$$

Applying GP regression requires  $\mathcal{O}(|\mathcal{S}|^3)$  time due to the matrix inversion in (1). We note that efficient approximation algorithms are available when working with long time series [Hensman et al., 2013; Quiñero-Candela and Rasmussen, 2005].

In the next section, we describe our proposed framework for defining kernels between time series based on Gaussian processes.

## 3 KERNELS FOR TIME SERIES

In this section, we first introduce the general mixture of sliding GP marginal kernels (MSM) framework for sparse and irregularly sampled data. We then introduce the expected Gaussian kernel, which serves as an uncertainty-aware base kernel within the MSM framework.

### 3.1 THE MIXTURE OF SLIDING GP MARGINAL KERNELS

As described in Section 2, we represent each time series  $\mathcal{S}$  in a data set  $\mathcal{D}$  using the posterior Gaussian process it induces under a GP regression model. The proposed mixture of sliding GP marginal kernels  $\mathcal{K}_{\text{MSM}}^{(d)}$  defines a kernel between a pair of time series through a weighted average of a base kernel  $\mathcal{K}_{\text{B}}$  applied to a collection of finite posterior GP marginals. Specifically, let  $u_1, \dots, u_L$  be a uniformly-spaced set of  $L$  time points on  $[0, T]$ , and  $\mathbf{u}^{(s)} = [u_s, \dots, u_{s+d-1}]^\top$  be a window of  $d$  time points starting at  $u_s$ . The MSM kernel compares the posterior GP marginals over the complete collection of valid sliding windows  $\mathbf{u}^{(1)}, \dots, \mathbf{u}^{(L-d+1)}$  as shown below, provided  $w_s \geq 0$  for all  $s$ .

$$\mathcal{K}_{\text{MSM}}^{(d)}(\mathcal{S}_i, \mathcal{S}_j) = \sum_{s=1}^{L-d+1} w_s \mathcal{K}_{\text{B}}\left(\mathcal{GP}(\mathbf{u}^{(s)} | \mathcal{S}_i), \mathcal{GP}(\mathbf{u}^{(s)} | \mathcal{S}_j)\right)$$

<sup>1</sup>See Rasmussen and Williams [2006] for details.

The length of the windows  $d$  is a hyper-parameter of the MSM kernel. In this work, we choose uniform kernel mixture weights  $w_s = 1/k$ . Alternatively, the kernel weights can be learned from data using multiple kernel learning algorithms [Bach et al., 2004].

The base kernel  $\mathcal{K}_B$  can be any valid kernel that takes as input two  $d$ -dimensional Gaussians. Of particular interest are uncertainty-aware base kernels that use the covariance information in the posterior marginals to modulate the similarity between the distributions. We present an uncertainty-aware expected Gaussian kernel in Section 3.3, but first describe a simpler kernel to highlight the trade-offs induced by the window length parameter  $d$ .

### 3.2 GAUSSIAN KERNEL ON MARGINAL MEANS

The Gaussian kernel  $\mathcal{K}_G$  below is one of the most widely used kernels in machine learning.

$$\mathcal{K}_G(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{1}{2\gamma^2}\|\mathbf{x}_i - \mathbf{x}_j\|^2\right) \quad (2)$$

The parameter  $\gamma$  controls the bandwidth of the kernel. The Gaussian kernel  $\mathcal{K}_G$  provides a simple kernel  $\mathcal{K}_{G\mu}$  between Gaussian distributions  $\mathcal{N}_i = \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$  and  $\mathcal{N}_j = \mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$  when applied to their mean vectors as follows.

$$\mathcal{K}_{G\mu}(\mathcal{N}_i, \mathcal{N}_j) = \mathcal{K}_G(\boldsymbol{\mu}_i, \boldsymbol{\mu}_j) \quad (3)$$

Importantly, this kernel is not uncertainty aware as it discards the covariances from the posterior Gaussians. We use the notation  $\mathcal{K}_{MG}^{(d)}$  to denote the use of  $\mathcal{K}_{G\mu}$  as the base kernel within the MSM framework. In the case where  $d = 1$ , the  $\mathcal{K}_{MG}^{(d)}$  kernel corresponds to taking the average similarity between the means of the two marginal posterior distributions as seen below.

$$\mathcal{K}_{MG}^{(1)}(\mathcal{S}_i, \mathcal{S}_j) = \frac{1}{L} \sum_{s=1}^L \exp\left(-\frac{1}{2\gamma^2}(\mu_{is} - \mu_{js})^2\right)$$

On the other hand, when  $d = L$ , the  $\mathcal{K}_{MG}^{(d)}$  kernel is equivalent to a product of the similarities between the means of the two marginal posterior distributions as seen below.

$$\begin{aligned} \mathcal{K}_{MG}^{(L)}(\mathcal{S}_i, \mathcal{S}_j) &= \exp\left(-\frac{1}{2\gamma^2} \sum_{s=1}^L (\mu_{is} - \mu_{js})^2\right) \\ &= \prod_{s=1}^L \exp\left(-\frac{1}{2\gamma^2} (\mu_{is} - \mu_{js})^2\right) \end{aligned}$$

This comparison shows that  $\mathcal{K}_{MG}^{(1)}$  is much more likely to be robust to the influence of noise and outliers due to the use of averaging, but it ignores the broader structure across time points. On the other hand,  $\mathcal{K}_{MG}^{(L)}$  captures the broader structure across time points, but may be more sensitive to

the presence of noise and outliers due to the product form of the kernel. Importantly, the MSM kernel framework is able to balance these considerations by allowing for the selection of intermediate window lengths  $d$ .

### 3.3 THE EXPECTED GAUSSIAN KERNEL

In this section, we present an uncertainty-aware base kernel  $\mathcal{K}_{EG}$ , which we refer to as the *expected Gaussian kernel*. This kernel is obtained as the expectation of the standard Gaussian kernel shown in (2) under the two independent Gaussians  $\mathcal{N}_i$  and  $\mathcal{N}_j$

$$\mathcal{K}_{EG}(\mathcal{N}_i, \mathcal{N}_j) = \mathbb{E}_{\mathbf{x}_i \sim \mathcal{N}_i, \mathbf{x}_j \sim \mathcal{N}_j} [\mathcal{K}_G(\mathbf{x}_i, \mathbf{x}_j)].$$

Importantly, the value of the expected Gaussian kernel can be computed analytically as shown in (4) where  $\tilde{\boldsymbol{\mu}} = \boldsymbol{\mu}_i - \boldsymbol{\mu}_j$  and  $\tilde{\boldsymbol{\Sigma}} = \boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_j + \gamma^2 \mathbf{I}$ . (see Appendix A for the derivation).

$$\mathcal{K}_{EG}(\mathcal{N}_i, \mathcal{N}_j) = \sqrt{\frac{|\boldsymbol{\Sigma}|}{|\tilde{\boldsymbol{\Sigma}}|}} \exp\left(-\frac{1}{2} \tilde{\boldsymbol{\mu}}^\top \tilde{\boldsymbol{\Sigma}}^{-1} \tilde{\boldsymbol{\mu}}\right). \quad (4)$$

The positive definiteness of the expected Gaussian kernel follows from the fact that the Gaussian kernel is positive definite and therefore there exists a map  $\phi$  such that the kernel acts as a dot product  $\langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$ . With the independence assumption, the expected Gaussian kernel also acts as a dot product over the expected map [Smola et al., 2007].

$$\begin{aligned} \mathcal{K}_{EG}(\mathcal{N}_i, \mathcal{N}_j) &= \mathbb{E}_{\mathbf{x}_i \sim \mathcal{N}_i, \mathbf{x}_j \sim \mathcal{N}_j} \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle \\ &= \langle \mathbb{E}_{\mathbf{x} \sim \mathcal{N}_i} [\phi(\mathbf{x})], \mathbb{E}_{\mathbf{x} \sim \mathcal{N}_j} [\phi(\mathbf{x})] \rangle. \end{aligned}$$

Interestingly, the probability product kernel of Jebara et al. [2004] applied to a pair of Gaussian distributions

$$\mathcal{K}_{PP}(\mathcal{N}_i, \mathcal{N}_j) = \int \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)^\rho \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)^\rho d\mathbf{x}$$

when  $\rho = 1$  (also known as the expected likelihood kernel) is a limiting case of the expected Gaussian kernel as  $\gamma \rightarrow 0$ . In this case, the  $\mathcal{K}_G$  term inside  $\mathcal{K}_{EG}$  degenerates to the Dirac delta function  $\delta(\mathbf{x}_i - \mathbf{x}_j)$ , and the expected Gaussian kernel collapses to the probability product kernel with  $\rho = 1$  by the sifting property of the delta function.

We refer to the use of the expected Gaussian kernel within the MSM framework as the *mixture of expected Gaussian kernels (MEG)*. Similar to  $\mathcal{K}_{MG}^{(d)}$ , the MEG kernel is able to strike a balance between the use of averaging to mitigate noise and the use of higher-dimensional marginals to capture broader temporal structure under uncertainty through the choice of  $d$ .

In terms of computational complexity, computing the expected Gaussian kernel (4) for  $d$ -dimensional Gaussians takes  $\mathcal{O}(d^3)$  time because of the inversion of  $\tilde{\boldsymbol{\Sigma}}$  and the



computation of its determinant. As a result, for the MEG kernel involving  $k$  GP marginals of  $d$  dimensions, it takes  $\mathcal{O}(kn^2d^3)$  time to compute the  $n \times n$  kernel matrix over  $n$  data cases. In the next section, we discuss scaling learning with MEG kernels to large data sets using random feature approximations.

## 4 RANDOM FOURIER FEATURES

The  $\mathcal{O}(n^2)$  kernel matrix computation time is a significant limitation when working with large data sets. Random Fourier feature approximation [Rahimi and Recht, 2007] is a kernel approximation algorithm based on Bochner’s theorem that maps the input data into a randomized low-dimensional feature space to approximate a shift-invariant kernel. In this section, we show how to extend this idea to scale-up learning with expected Gaussian kernels and apply the result to the MEG kernel.

### 4.1 RANDOM FEATURES FOR EXPECTED GAUSSIAN KERNELS

Following the construction presented by Rahimi and Recht [2007], the Gaussian kernel  $\mathcal{K}_G$  defined in (2) can be approximated by an  $m$ -dimensional random vector

$$\mathbf{z}(\mathbf{x}) = \sqrt{\frac{2}{m}} \left[ \cos(\mathbf{w}_1^\top \mathbf{x} + b_1), \dots, \cos(\mathbf{w}_m^\top \mathbf{x} + b_m) \right]^\top,$$

where  $\mathbf{w}_i \sim \mathcal{N}(\mathbf{0}, \gamma^{-2}\mathbf{I})$ ,<sup>2</sup> and  $b_i \sim \text{uniform}(0, 2\pi)$  so that  $\mathcal{K}_G(\mathbf{x}_i, \mathbf{x}_j) \approx \mathbf{z}(\mathbf{x}_i)^\top \mathbf{z}(\mathbf{x}_j)$ .

The analytic form of the expected Gaussian kernel given in (4) is not shift invariant in terms of the means and covariances of the input Gaussians, and therefore we cannot directly expand the kernel as in Rahimi and Recht [2007]. However, we can derive the random Fourier features for the expected Gaussian kernel by taking the expectation after Gaussian kernel expansion. With the independence of the input Gaussians, we have

$$\begin{aligned} \mathbb{E}_{\mathbf{x}_i, \mathbf{x}_j} [\mathcal{K}_G(\mathbf{x}_i, \mathbf{x}_j)] &\approx \mathbb{E}_{\mathbf{x}_i, \mathbf{x}_j} [\mathbf{z}(\mathbf{x}_i)^\top \mathbf{z}(\mathbf{x}_j)] \\ &= \mathbb{E}_{\mathbf{x}_i} [\mathbf{z}(\mathbf{x}_i)]^\top \mathbb{E}_{\mathbf{x}_j} [\mathbf{z}(\mathbf{x}_j)]. \end{aligned}$$

Next, we note that each entry of  $\mathbb{E}_{\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})} [\mathbf{z}(\mathbf{x})]$  can be obtained analytically as shown below. This result exploits the fact that the expectation of the complex random feature map  $\exp(i\mathbf{w}^\top \mathbf{x})$  derived from the Fourier expansion of the kernel function is the characteristic function of the distribution of  $\mathbf{x}$ . A detailed derivation is given in Appendix B.

$$\begin{aligned} \mathbb{E}[z_i(\mathbf{x})] &= \sqrt{\frac{2}{m}} \mathbb{E}_{\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})} [\cos(\mathbf{w}_i^\top \mathbf{x} + b_i)] \\ &= \sqrt{\frac{2}{m}} \exp\left(-\frac{1}{2} \mathbf{w}_i^\top \boldsymbol{\Sigma} \mathbf{w}_i\right) \cos(\mathbf{w}_i^\top \boldsymbol{\mu} + b_i). \end{aligned}$$

<sup>2</sup>  $\mathbf{w}_i \sim \mathcal{N}(\mathbf{0}, \gamma^{-2}\mathbf{I})$  can be done with each entry drawn independently from  $\mathcal{N}(0, \gamma^{-2})$ .

---

### Algorithm 1: Random Fourier Features for $\mathcal{K}_{EG}$

---

**Input:** A Gaussian  $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$  with mean  $\boldsymbol{\mu}$  and covariance  $\boldsymbol{\Sigma}$ . Width parameter  $\gamma^2$  of the Gaussian kernel.

Number of random features  $m$ .

$\mathbf{w}_1, \dots, \mathbf{w}_m \stackrel{\text{iid}}{\sim} \mathcal{N}(\mathbf{0}, \gamma^{-2}\mathbf{I})$

$b_1, \dots, b_m \stackrel{\text{iid}}{\sim} \text{uniform}(0, 2\pi)$

**return**  $\sqrt{\frac{2}{m}} \begin{bmatrix} \exp\left(-\frac{1}{2} \mathbf{w}_1^\top \boldsymbol{\Sigma} \mathbf{w}_1\right) \cos(\mathbf{w}_1^\top \boldsymbol{\mu} + b_1) \\ \vdots \\ \exp\left(-\frac{1}{2} \mathbf{w}_m^\top \boldsymbol{\Sigma} \mathbf{w}_m\right) \cos(\mathbf{w}_m^\top \boldsymbol{\mu} + b_m) \end{bmatrix}$

---

As we can see, each random feature for an expected Gaussian kernel is the product of  $\sqrt{2/m} \cos(\mathbf{w}_i^\top \boldsymbol{\mu} + b_i)$  and  $\exp\left(-\frac{1}{2} \mathbf{w}_i^\top \boldsymbol{\Sigma} \mathbf{w}_i\right)$ . The former is identical to the random Fourier feature with the Gaussian mean as input. The latter is an exponential decay term that decreases as uncertainty in the distribution increases.

The complete procedure for obtaining a random features approximation for the expected Gaussian kernel is given in Algorithm 1.

For the  $d$ -dimensional case, approximating an expected Gaussian kernel with  $m$  random features using Algorithm 1 requires  $\mathcal{O}(md^2)$  time as it takes  $\mathcal{O}(d^2)$  time to compute the quadratic term  $\mathbf{w}^\top \boldsymbol{\Sigma} \mathbf{w}$ . As a result, given a data set of size  $n$ , it takes  $\mathcal{O}(nmd^2)$  to compute the  $n \times m$  feature matrix. This is more efficient compared to the  $\mathcal{O}(n^2d^3)$  time needed to compute the exact kernel, especially when  $n \gg m$ .

### 4.2 ACCELERATION FOR HIGH-DIMENSIONAL GAUSSIANS

The  $\mathcal{O}(d^2)$  time for computing the random features can be computationally prohibitive when working with high-dimensional Gaussians. Le et al. [2013] proposed Fastfood to accelerate the computation of the original random Fourier features of Rahimi and Recht [2007]. Fastfood utilizes the fast Walsh-Hadamard transform to simulate a full Gaussian random matrix using a small portion of i.i.d. Gaussian samples. Essentially, given a vector  $\mathbf{x} \in \mathbb{R}^d$ , Fastfood approximates the matrix-vector product  $\mathbf{V}\mathbf{x}$  in  $\mathcal{O}(m \log d)$  time instead of  $\mathcal{O}(md)$ , where  $\mathbf{V}$  is an  $m \times d$  random matrix with each entry drawn independently from  $\mathcal{N}(0, \gamma^{-2})$ .

With Fastfood, computing the  $\cos(\mathbf{w}_i^\top \boldsymbol{\mu} + b_i)$  term for the expected Gaussian kernel for all  $i = 1, \dots, m$  can be done by first generating the random vector  $\mathbf{V}\boldsymbol{\mu}$  as described above. All  $m$  entries  $\cos([\mathbf{V}\boldsymbol{\mu}]_i + b_i)$  can then be computed in  $\mathcal{O}(m \log d)$  time.

The bottleneck for the expected Gaussian kernel is the computation of the exponential term  $\exp\left(-\frac{1}{2} \mathbf{w}_i^\top \boldsymbol{\Sigma} \mathbf{w}_i\right)$ , which needs  $\mathcal{O}(d^2)$  time if computed naively. This can also be

accelerated by using the Fastfood trick twice:

$$\exp\left(-\frac{1}{2}\mathbf{w}_i^\top \boldsymbol{\Sigma} \mathbf{w}_i\right) = \exp\left(-\frac{1}{2}[\mathbf{V}(\mathbf{V}\boldsymbol{\Sigma})^\top]_{ii}\right). \quad (5)$$

Following the stacking strategy in Le et al. [2013], we choose  $\mathbf{V}$  in (5) to be a  $d \times d$  square matrix<sup>3</sup>, and repeat this step  $\lceil m/d \rceil$  times to produce all  $m$  features. This leads to an overall cost of  $\mathcal{O}(md \log d)$  as opposed to the  $\mathcal{O}(md^2)$  time mentioned before to compute  $m$  random features for the expected Gaussian kernel taking on a single  $d$ -dimensional Gaussian input.

We can further reduce the cost by approximating the covariance matrix with a low rank matrix  $\boldsymbol{\Sigma} \approx \boldsymbol{\Phi}\boldsymbol{\Phi}^\top$  using truncated SVD where  $\boldsymbol{\Phi} \in \mathbb{R}^{d \times r}$ . There exists efficient algorithms to compute top- $r$  SVD such as randomized SVD [Halko et al., 2011] that requires  $\mathcal{O}(d^2 \log r)$  time in contrast with  $\mathcal{O}(d^3)$  for classical algorithms. With Fastfood, the exponential term can be approximated in  $\mathcal{O}(r \log d)$  time:

$$\exp\left(-\frac{1}{2}\mathbf{w}_i^\top \boldsymbol{\Sigma} \mathbf{w}_i\right) \approx \exp\left(-\frac{1}{2}\sum_{j=1}^r [\mathbf{V}\boldsymbol{\Phi}]_{ij}^2\right). \quad (6)$$

This leads to  $\mathcal{O}(mr \log d)$  time for some  $r < d$  to compute  $m$  random features for a single data case.

### 4.3 RANDOM FEATURES FOR THE MIXTURE OF EXPECTED GAUSSIAN KERNELS

Let  $\mathbf{z}(\mathcal{N})$  denote the random features for the expected Gaussian kernel computed by Algorithm 1. As described in Section 3.1, each time series  $\mathcal{S}_i$  is summarized by a collection of  $k$  Gaussian marginals  $\{\mathcal{N}_i^{(1)}, \dots, \mathcal{N}_i^{(k)}\}$  for all  $i$ . The mixture of expected Gaussian kernels can be approximated by the random features of the base kernel applied to each marginal:

$$\begin{aligned} \sum_{s=1}^k w_s \mathcal{K}_{\text{EG}}(\mathcal{N}_i^{(s)}, \mathcal{N}_j^{(s)}) &\approx \sum_s w_s \mathbf{z}_s(\mathcal{N}_i^{(s)})^\top \mathbf{z}_s(\mathcal{N}_j^{(s)}) \\ &= \sum_s \left(\sqrt{w_s} \mathbf{z}_s(\mathcal{N}_i^{(s)})\right)^\top \left(\sqrt{w_s} \mathbf{z}_s(\mathcal{N}_j^{(s)})\right). \end{aligned}$$

Equivalently, each time series can be expressed as the compound random feature map below to approximate the MEG kernel.

$$\widehat{\mathbf{z}}(\mathcal{S}) = \left[\sqrt{w_1} \mathbf{z}_1(\mathcal{N}^{(1)})^\top, \dots, \sqrt{w_k} \mathbf{z}_k(\mathcal{N}^{(k)})^\top\right]^\top. \quad (7)$$

In this work, we set  $w_s = 1/k$  for all  $k$  marginals, that is, the base kernels are weighted equally. Furthermore, each marginal  $\mathcal{N}^{(s)}$  is approximated by the same number of random features  $m$ . Therefore,  $\widehat{\mathbf{z}}(\mathcal{S})$  has  $mk$  random features

<sup>3</sup> Assume  $\boldsymbol{\Sigma}$  is properly padded so that the dimension becomes  $d = 2^\ell$  in order to perform Hadamard transform [Le et al., 2013].

in total. In Section 4.4, we will show that having the same number of random features for each marginal will lead to the lowest error bound under uniform weights.

In general,  $w_s$  can be the coefficients of any non-negative combination, either chosen according to domain knowledge or learned from data. Learning the weights from data with the random features given in (7) can be viewed as an approximation to multiple kernel learning [Bach et al., 2004]. Optimizing  $w_1, \dots, w_k$  is similar to Mahalanobis metric learning [Xing et al., 2002] for the diagonal case except that all random features come from the same base kernel share a scaling factor.

### 4.4 APPROXIMATION GUARANTEES

In this section, we analyze the approximation quality of the expected Gaussian kernel random features computed by Algorithm 1 in terms of the concentration of the approximating kernel matrix. Using the Hermitian matrix Bernstein inequality [Tropp, 2012a,b] and following a derivation similar to [Lopez-Paz et al., 2014], we can bound the spectral norm (denoted  $\|\cdot\|$ ) of the difference between the exact expected Gaussian kernel and its approximation.<sup>4</sup>

**Theorem 1.** *Given a data set with each example represented as a single Gaussian,  $\mathcal{N}_1, \dots, \mathcal{N}_n$ , let  $\mathbf{K} \in \mathbb{R}^{n \times n}$  be the expected Gaussian kernel matrix. Let  $\widehat{\mathbf{K}} \in \mathbb{R}^{n \times n}$ , with each entry  $[\widehat{\mathbf{K}}]_{ij} = \mathbf{z}(\mathcal{N}_i)^\top \mathbf{z}(\mathcal{N}_j)$ , be the approximation matrix constructed using Algorithm 1 with  $m$  random features. Then we have*

$$\mathbb{E}\|\widehat{\mathbf{K}} - \mathbf{K}\| \leq \frac{2n}{m} \sqrt{\frac{2 \log n}{m}} + \frac{2n \log n}{3m^2}.$$

The proof of Theorem 1 is given in Appendix C. It states that the error  $\mathbb{E}\|\widehat{\mathbf{K}} - \mathbf{K}\|$  is bounded by  $\mathcal{O}(n \log n)$  for  $n$  data cases with a fixed number of random features  $m$ . On the other hand, for a fixed number of data cases  $n$ , increasing the number of random features  $m$  induces an  $\mathcal{O}(m^{-3/2})$  reduction in error.

As for the high-dimensional case described in Section 4.2, Le et al. [2013] have shown that the Fastfood feature map is unbiased, and therefore Theorem 1 also holds for the random features computed by Fastfood using (5). However, with the low-rank approximation used in (6),  $\widehat{\mathbf{K}}$  no longer converges to  $\mathbf{K}$  but instead converges to  $\widetilde{\mathbf{K}}$  where

$$[\widetilde{\mathbf{K}}]_{ij} = \mathcal{K}_{\text{EG}}\left(\mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Phi}_i \boldsymbol{\Phi}_i^\top), \mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Phi}_j \boldsymbol{\Phi}_j^\top)\right).$$

Following the construction described in Section 4.3, the mixture of  $k$  expected Gaussian kernels has a total of  $M = |\widehat{\mathbf{z}}(\mathcal{S})| = mk$  features. Since  $w_s = 1/k$  for all  $s$ , each entry of the feature vector is in the form of

<sup>4</sup> This bound can also be applied to the original random Fourier feature approximation [Rahimi and Recht, 2007].

$\sqrt{2/M} \mathbb{E}[\cos(\mathbf{w}^\top \mathbf{x} + b)]$ , whose absolute value is bounded by  $\sqrt{2/M}$ . Following the proof of Theorem 1, we can bound the error of using the proposed random feature approximation to the MEG kernel.

**Corollary 1.** Consider the MEG kernel consisting of  $k$  base kernels. Let  $\mathbf{K} \in \mathbb{R}^{n \times n}$  be the MEG kernel matrix, and  $\widehat{\mathbf{K}}$  be the approximating matrix using  $M = mk$  random features. Then,

$$\mathbb{E} \|\widehat{\mathbf{K}} - \mathbf{K}\| \leq \frac{2n}{M} \sqrt{\frac{2 \log n}{M}} + \frac{2n \log n}{3M^2}. \quad (8)$$

The expected error bound in Corollary 1 has the same form as that in Theorem 1 except the bound is determined by the number of total random features  $M$ . When the number of kernels  $k$  is large, even if each kernel is approximated by only a few random features, a low error bound can still be achieved if  $M = mk$  is sufficiently large.

As a matter of fact, for a *convex combination* of  $k$  expected Gaussian kernels with unequal weights, choosing the number of random features proportional to the corresponding kernel weight will achieve an error bound identical to (8) for a total of  $M$  random features.

## 5 EXPERIMENTS

We evaluate the proposed MEG kernel and the corresponding random feature approximation in terms of time series classification in the presence of sparse and irregular sampling. In the sections below, we describe the experimental methodology and the results.

### 5.1 EXPERIMENTAL METHODOLOGY

**Data.** We conduct experiments on all 43 time series data sets from the UCR time series classification archive [Keogh et al., 2011]. The UCR archive contains a diverse collection of time series data sets that vary significantly in terms of length, number of classes, and number of data cases. However, all the data sets are densely and uniformly sampled. This allows us to perform controlled experiments where we decrease the sampling density and observe the effect on the relative performance of different classification methods. We consider ten different sampling densities from 10% to 100% in steps of 10%.

**Gaussian Process Representation.** Following Section 2, for each data set and each sampling density, we first learn the hyperparameters of the GP regression model by optimizing the log marginal likelihood over the observed data. As described in Section 3.1, we compute the posterior GP marginals over a uniformly-spaced grid of  $L$  points on  $[0, T]$ , where  $T$  is the common length of the fully observed time series of each data set. We select  $L = \min(3T, 500)$ .

**Kernel and Feature Normalization.** We apply standard

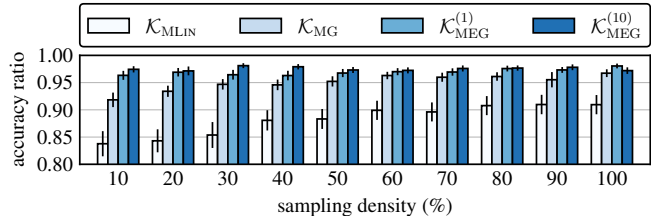


Figure 1: Comparing MSM kernel framework with different base kernels.

kernel normalization to all kernel matrices before averaging. We also normalize each random feature vector to have unit length. Empirically, we found that normalization improves the classification performance.

**Base Classifiers and Hyperparameters.** We use support vector machines (SVMs) for classification. For kernel-based methods we use libsvm [Chang and Lin, 2011] with precomputed kernels. For random feature approximation, we use liblinear [Fan et al., 2008], which is tailored for linear models. We use five-fold stratified cross validation to jointly select the SVM regularization parameter and the parameter  $\gamma$  for the expected Gaussian kernels.

**Accuracy Measures.** We assess the performance of each method in terms of classification accuracy using the benchmark train/test splits in the UCR archive. We report results in terms of average *accuracy ratios* over all 43 data sets to emphasize the relative differences between methods across different sampling densities. For a given data set and sampling density, the accuracy ratio for a method is the accuracy of the method divided by the accuracy of the best performing method on that data set and sampling density. We also report one-standard-error error bars.

### 5.2 EXPERIMENTS AND RESULTS

**Comparing Base Kernels for MSM Framework.** We evaluate several instances of the MSM framework using different base kernels. The linear MSM kernel  $\mathcal{K}_{\text{MLIN}}$  uses the linear kernel on the univariate marginal means  $\mathcal{K}_{\text{LIN}}(\mathcal{N}_i, \mathcal{N}_j) = \mu_i \mu_j$  as the base kernel. The Gaussian MSM kernel  $\mathcal{K}_{\text{MG}}$  uses  $\mathcal{K}_{G\mu}$  defined in (3) also on the univariate marginal means. We compare these baseline methods to two expected Gaussian kernel based MSM kernels: the MEG kernel  $\mathcal{K}_{\text{MEG}}^{(1)}$  on the univariate marginals, and the MEG kernel  $\mathcal{K}_{\text{MEG}}^{(10)}$  with a sliding window size of 10.

Figure 1 shows the classification performance of these methods on each sampling density. The Gaussian MSM kernel  $\mathcal{K}_{\text{MG}}$  significantly outperforms the linear MSM kernel  $\mathcal{K}_{\text{MLIN}}$ . However,  $\mathcal{K}_{\text{MEG}}^{(1)}$  and  $\mathcal{K}_{\text{MEG}}^{(10)}$  both outperform  $\mathcal{K}_{\text{MG}}$ , particularly under high sparsity. This is expected since  $\mathcal{K}_{\text{MEG}}^{(1)}$  and  $\mathcal{K}_{\text{MEG}}^{(10)}$  both capture posterior uncertainty while  $\mathcal{K}_{\text{MG}}$  does not.

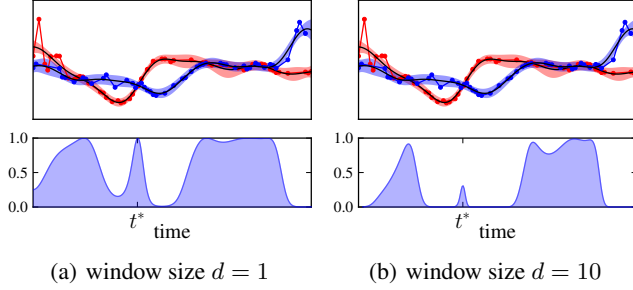


Figure 2: Comparison of different window size  $d$ . The plot on the top of each panel shows two time series from the ECG200 data set at 50% sampling density with visualization of their posterior Gaussian process. The plot on the bottom shows the value of the corresponding expected Gaussian kernel at each time slice.

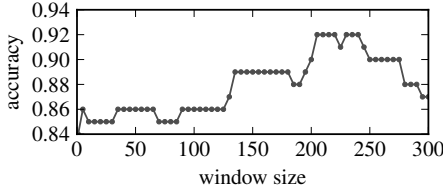


Figure 3: Comparison of classification accuracy under different window sizes on ECG200 at 50% sampling density.

**Effect of Sliding Window Size for MEG Kernel.** Figure 2 illustrates how different window sizes affect the similarity output by the expected Gaussian kernel on the ECG200 data set from the UCR archive at 50% sampling density. The two time series intersect at around  $t^*$ ; however, they have opposite trends at this time point. Since  $\mathcal{K}_{\text{MEG}}^{(1)}$  does not take local correlation structure into account at all, it achieves the highest possible value at  $t^*$ . On the other hand,  $\mathcal{K}_{\text{MEG}}^{(10)}$  outputs a low value at  $t^*$  since a larger window captures the fact that the two processes are anti-correlated in the neighborhood of  $t^*$ .

Figure 3 shows the classification performance across various window sizes ranging from 1 to  $L$  on the ECG200 data set at 50% sampling density. For this experiment, we fixed the SVM regularization parameter to  $C = 2000$ , and the covariance parameter of the expected Gaussian kernel to  $\gamma = 0.01d$ , which grows linearly as the window size  $d$  increases. Empirically, such choice of  $\gamma$  makes the values of the expected Gaussian kernels numerically stable.

The results show that the classification accuracy on ECG200 improves as the window size increases and peaks at around  $0.75L$ . We note that using larger window size is computationally more expensive, and that not all data sets show a benefit with increasing window size.

**Comparing MEG to Existing Methods.** We compare the MEG kernel with two existing methods for time series classification. The reproducing kernel Hilbert space (RKHS)

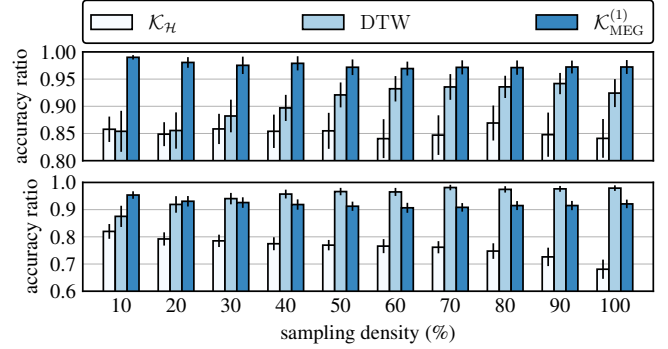


Figure 4: Comparison with other time series classification methods. The plot on the top corresponds to the 20 data sets whose optimal warping window size is at most 12; the plot on the bottom corresponds to the rest of 23 data sets with optimal warping window size greater than 12.

kernel  $\mathcal{K}_{\mathcal{H}}$  proposed by Lu et al. [2008] is a time series kernel also designed for irregularly sampled time series. The RKHS kernel is defined as the squared norm between two posterior GP mean functions in the RKHS induced by a common prior covariance function. The kernel can be computed in closed form, but also discards posterior uncertainty since it only depends on the posterior GP means. It is also not possible to focus the kernel on assessing similarity over a specific time interval.

Dynamic time warping [Berndt and Clifford, 1994; Sakoe and Chiba, 1971] (DTW) is a widely used distance function for misaligned and warped time series. We compare to the 1-nearest neighbor algorithm using DTW distance subject to the Sakoe-Chiba warping constraint using the optimal window size published along with the UCR time series archive. Since classic DTW is not designed for irregularly sampled time series data, we also use GP regression to interpolate each time series on the same set of reference time points as the MSM kernels, and use the posterior means as the input to DTW.

In this experiment, we split the data sets into two groups according to their published optimal warping window sizes. Smaller warping window size implies the corresponding time series are almost aligned and have minimal warping. The 23 data sets with optimal window size greater than 12 are selected as the *warped* group, which implies that the corresponding time series require significant alignment or warping before direct comparison. The need for alignment and warping violates the assumptions of the MSM kernel as well as the RKHS kernel, which does not explicitly take warping or alignment into account. The rest of the 20 data sets are regarded as the *aligned* group.

Figure 4 shows that the RKHS kernel  $\mathcal{K}_{\mathcal{H}}$  consistently performs the worst on both groups, because it fails to focus on a finite time interval of interest where data points are observed and does not account for uncertainty. For

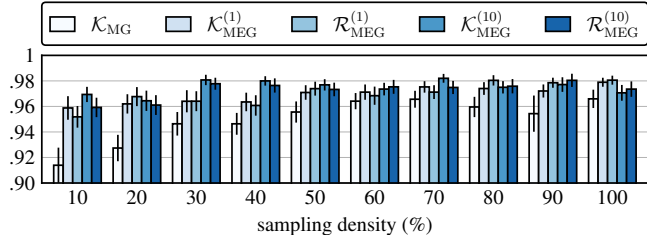


Figure 5: Comparison of the classification accuracy ratio of the exact time series expected Gaussian kernel against the random Fourier feature approximation. The baseline method  $\mathcal{K}_{MG}$  is included as a reference.

the aligned group, our method always outperforms DTW. When the time series is more sparsely sampled, the advantage of our uncertainty-aware framework becomes more significant. For the warped group, DTW achieves better classification accuracy under low sparsity, but our approach achieves comparable or better accuracy under high sparsity while the RKHS kernel does not.

**Random Features for MEG Kernels.** To evaluate the random feature approximation to the MEG kernel in terms of classification accuracy, we use  $m = \lceil 10,000/k \rceil$  random features to approximate the expected Gaussian kernel on each of  $k$  marginals, so that the total number of random features  $M = mk$  is at most 10,000. In the experiment,  $\mathcal{R}_{MEG}^{(1)}$  and  $\mathcal{R}_{MEG}^{(10)}$  denote the random feature approximation for  $\mathcal{K}_{MEG}^{(1)}$  and  $\mathcal{K}_{MEG}^{(10)}$  with window size 1 and 10.

Figure 5 shows that the random feature approximation produces similar classification results compared to the exact kernels for both marginal window sizes. The baseline method  $\mathcal{K}_{MG}$  is included to show that even when the accuracy declines due to approximation, it still outperforms the baseline method.

Table 1 shows the classification training and prediction time on the four largest data sets in the UCR archive. We divide the training and prediction task using either  $\mathcal{K}_{MEG}^{(1)}$  or  $\mathcal{R}_{MEG}^{(1)}$  into two steps: first, computing the kernel matrix for  $\mathcal{K}_{MEG}^{(1)}$  or the random feature matrix for  $\mathcal{R}_{MEG}^{(1)}$ , which are shown as the 3rd and 5th column (denoted prep.) in Table 1; second, training and prediction time spent solely in the classifier, denoted train and test in the table.

The results in Table 1 show that computing the feature/kernel matrix dominates the entire training/prediction task. It is consistent with the time and space complexity analysis given in Table 2. In terms of the data size, computing the exact kernel takes  $\mathcal{O}(n^2)$  time, while computing the random feature matrix takes  $\mathcal{O}(n)$  time. As the window size  $d$  increases, computing the exact kernel takes  $\mathcal{O}(d^3)$  time as oppose to  $\mathcal{O}(d^2)$  for random feature approximation.

As for the actual classifier learning and prediction time, we can see that  $\mathcal{R}_{MEG}^{(1)}$  takes longer than  $\mathcal{K}_{MEG}^{(1)}$ . This is because

Table 1: Comparison of classification time (in seconds) on the four largest data sets using exact expected Gaussian kernels as opposed to random feature approximation under window sizes 1 and 10. In the table, the MEG kernel subscripts are dropped from the notation for brevity. The two numbers  $(n, L)$  for each data set denote the number of examples and the number of reference time points. Note that a MEG kernel with window size  $d$  consists of  $k = L - d + 1$  base kernels (see Section 3).

| data                      | meth.                | prep.    | train | prep.    | test |
|---------------------------|----------------------|----------|-------|----------|------|
| TwoLead.<br>(1162, 246)   | $\mathcal{K}^{(1)}$  | 13.88    | 0.01  | 13.76    | 0.00 |
|                           | $\mathcal{R}^{(1)}$  | 0.91     | 1.44  | 0.37     | 0.01 |
|                           | $\mathcal{K}^{(10)}$ | 754.05   | 0.01  | 743.41   | 0.00 |
| yoga<br>(3300, 500)       | $\mathcal{R}^{(10)}$ | 10.73    | 1.91  | 5.32     | 0.01 |
|                           | $\mathcal{K}^{(1)}$  | 143.11   | 0.25  | 144.67   | 0.01 |
|                           | $\mathcal{R}^{(1)}$  | 2.30     | 33.56 | 1.09     | 0.02 |
| wafer<br>(7164, 456)      | $\mathcal{K}^{(10)}$ | 10751.85 | 0.32  | 10620.35 | 0.01 |
|                           | $\mathcal{R}^{(10)}$ | 50.44    | 31.08 | 12.86    | 0.02 |
|                           | $\mathcal{K}^{(1)}$  | 650.98   | 0.15  | 652.41   | 0.03 |
| StarLight.<br>(9236, 500) | $\mathcal{R}^{(1)}$  | 4.88     | 8.65  | 2.27     | 0.06 |
|                           | $\mathcal{K}^{(10)}$ | 50452.40 | 0.17  | 49159.89 | 0.05 |
|                           | $\mathcal{R}^{(10)}$ | 58.76    | 45.13 | 29.71    | 0.09 |
| StarLight.<br>(9236, 500) | $\mathcal{K}^{(1)}$  | 1103.91  | 0.21  | 1119.40  | 0.05 |
|                           | $\mathcal{R}^{(1)}$  | 5.97     | 55.42 | 2.63     | 0.12 |
|                           | $\mathcal{K}^{(10)}$ | 86676.10 | 0.46  | 83357.72 | 0.15 |
|                           | $\mathcal{R}^{(10)}$ | 99.30    | 17.44 | 35.26    | 0.11 |

the final kernel matrix for  $\mathcal{K}_{MEG}^{(1)}$  can be stored in  $\mathcal{O}(n^2)$  space, as oppose to  $\mathcal{O}(nmk)$  for  $\mathcal{R}_{MEG}^{(1)}$ , which is notably larger for our choice of  $m$  ( $mk \approx 10,000$ ). By adjusting  $m$ , the total time using  $\mathcal{R}_{MEG}^{(1)}$  can be further reduced, but it is already significantly faster overall with the value of  $m$  used here.

**Comparing Random Features to Nyström Method.** The Nyström method [Williams and Seeger, 2001] is a commonly used kernel approximation algorithm based on low-rank approximation to the full kernel matrix computed using a subset of the training data. We compare the time versus kernel approximation error trade-off when approximating the MEG kernel by the Nyström method and random features using window sizes 1 and 10. The experiment is conducted on the largest data set in the UCR archive, StarLightCurves, at 10% sampling density. For Nyström method, we plot the results using  $s = 10, 20, \dots, 500$  samples. For the random feature approximation, we plot the results using  $m = 1, 2, \dots, 100$  random features for each expected Gaussian kernel (at most 50,000 total features for the largest  $m$ ). Note that the number of training cases used for the Nyström method is at most the size of the training data; however, the number of random features can exceed the size of the training data.

The results show that the Nyström approximation can achieve higher approximation accuracy than random features when sufficient training data samples are used [Yang

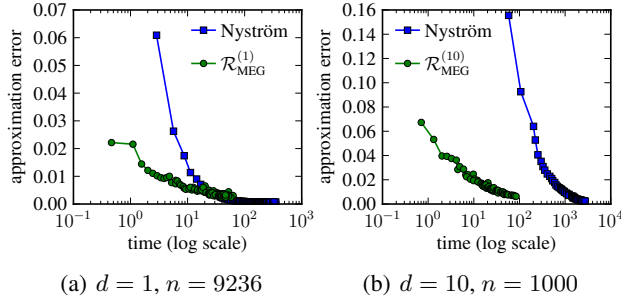


Figure 6: Comparing time versus approximation error (in terms of the relative error  $\mathbb{E}\|\tilde{\mathbf{K}} - \mathbf{K}\|/\|\mathbf{K}\|$ ) of Nyström method under various sizes of the subset of training data and random feature approximation with various numbers of random features.

Table 2: Comparing time and space complexity of the classification using the MEG kernel with window size  $d$  constructed from  $k$  expected Gaussian kernels, where  $n$  is the training data size,  $n'$  is the test data size. Nyström method uses a subset of the training data of size  $s$ , and  $\mathcal{R}_{\text{MEG}}^{(d)}$  uses a total of  $M = mk$  random features.

|             | $\mathcal{K}_{\text{MEG}}^{(d)}$ | Nyström                            | $\mathcal{R}_{\text{MEG}}^{(d)}$ |
|-------------|----------------------------------|------------------------------------|----------------------------------|
| train time  | $\mathcal{O}(d^3 n^2 k)$         | $\mathcal{O}(d^3 n s k + s^3)$     | $\mathcal{O}(d^2 n M)$           |
| test time   | $\mathcal{O}(d^3 n' n k)$        | $\mathcal{O}(d^3 n' s k + n' s^2)$ | $\mathcal{O}(d^2 n' M)$          |
| train space | $\mathcal{O}(n^2)$               | $\mathcal{O}(n s)$                 | $\mathcal{O}(n M)$               |
| test space  | $\mathcal{O}(n' n)$              | $\mathcal{O}(n' s)$                | $\mathcal{O}(n' M)$              |

et al., 2012]. However, for  $d$ -dimensional Gaussians (for the MEG kernel with window size  $d$ ), computing a single entry of the expected Gaussian kernel takes  $\mathcal{O}(d^3)$  comparing to  $\mathcal{O}(d^2)$  to compute a single random feature, as in the case of comparing to exact kernel computation. The detailed complexity analysis is given in Table 2. Figure 6(b) shows that for window size 10, the random feature approximation needs significantly less time to achieve an acceptable error rate.

**Fastfood Method for High-Dimensional Marginals.** We compare the straightforward random feature computation to two acceleration methods using Fastfood as described in Section 4.2. This experiment is conducted on the data set StarLightCurves from the UCR archive at 10% sampling density with the full window size  $L = 500$ . That is, there is a single expected Gaussian kernel with  $d = 500$  in the MEG kernel. We use randomized truncated SVD [Halko et al., 2011] for the low-rank approximation of covariance matrices with rank  $r = 10$ .

Figure 7 shows the time-versus-error relationship using three different methods with  $2^\ell$  random features for  $\ell = 9, \dots, 13$  (from top to bottom). It shows that all three methods achieve similar errors (relative error in terms of spectral norms) when using the same feature size. However, the Fastfood method using (5), denoted Fastfood in the figure, is at least 14 times faster among five feature sizes than the

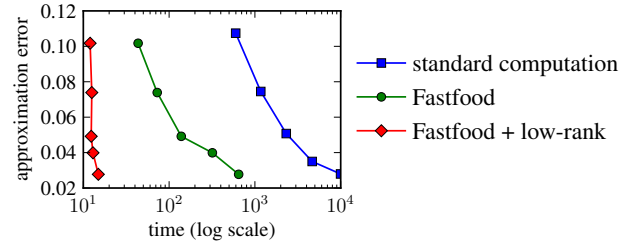


Figure 7: Comparing time versus approximation error (relative error) of the standard feature computation and the two Fastfood methods under  $d = 500$  and  $n = 1000$ . The five points for each method correspond to using 512, 1024, 2048, 4096, and 8192 features from top to bottom.

standard method, due to the  $\mathcal{O}(n M d \log d)$  time for Fastfood as opposed to  $\mathcal{O}(n M d^2)$ . With low-rank covariance approximation, the running time can be improved significantly again, even if an extra truncated SVD is required. The SVD overhead is roughly a constant of 2.7 seconds, which accounts for 86% time in the smallest case (512 features) and 42% in the largest case (8192 features).

## 6 CONCLUSIONS AND FUTURE WORK

We have proposed a kernel-based framework for classification of sparse and irregularly sampled time series that re-represents time series using Gaussian process and then assesses the similarity between the GPs based on the similarity between their finite marginals defined over sliding time windows. Our results show that the proposed approach achieves better average accuracy on a large time series classification benchmark compared to all other methods considered when the time series are aligned or under high sparsity. Further, our extension to random Fourier features achieves significant speedups relative to exact kernel computations as well as Nyström approximation on large time series data sets. Our application of Fastfood and low-rank covariance approximations yields further speedups in the case where large-dimensional marginals are required.

Possible directions for future work include learning kernel combination weights, extending the MSM framework to multi-output Gaussian processes for multivariate time series, and the extension of this framework to distributions other than Gaussians or different base kernels. Moreover, instead of performing classification on the random features using linear SVMs, we can use the random feature vector as an uncertainty-aware embedding of the data in various deep learning architectures, as well as unsupervised learning models for problems like clustering.

### Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. 1350522.

## References

- Bach, F. R., Lanckriet, G. R., and Jordan, M. I. (2004). Multiple kernel learning, conic duality, and the smo algorithm. In *Proceedings of the twenty-first international conference on Machine learning*, page 6. ACM.
- Berndt, D. J. and Clifford, J. (1994). Using dynamic time warping to find patterns in time series. In *KDD workshop*. Seattle, WA.
- Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- Clark, J. and Bjørnstad, O. (2004). Population time series: process variability, observation errors, missing values, lags, and hidden states. *Ecology*, 85(11):3140–3150.
- Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3):273–297.
- Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J. (2008). LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.
- Halko, N., Martinsson, P.-G., and Tropp, J. A. (2011). Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288.
- Hensman, J., Fusi, N., and Lawrence, N. D. (2013). Gaussian processes for big data. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence, Bellevue, WA, USA, August 11-15, 2013*.
- Jebara, T., Kondor, R., and Howard, A. (2004). Probability product kernels. *The Journal of Machine Learning Research*, 5:819–844.
- Keogh, E., Xi, X., Wei, L., and Ratanamahatana, C. A. (2011). The UCR time series classification/clustering homepage: [www.cs.ucr.edu/~eamonn/time\\_series\\_data/](http://www.cs.ucr.edu/~eamonn/time_series_data/).
- Le, Q., Sarló, T., and Smola, A. (2013). Fastfood—approximating kernel expansions in loglinear time. In *ICML*.
- Lopez-Paz, D., Sra, S., Smola, A. J., Ghahramani, Z., and Schölkopf, B. (2014). Randomized nonlinear component analysis. In *ICML*.
- Lu, Z., Leen, T. K., Huang, Y., and Erdogmus, D. (2008). A reproducing kernel hilbert space framework for pairwise time series distances. In *Proceedings of the 25th international conference on Machine learning*, pages 624–631. ACM.
- Marlin, B. M., Kale, D. C., Khemani, R. G., and Wetzel, R. C. (2012). Unsupervised pattern discovery in electronic health care data using probabilistic clustering models. In *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium*, pages 389–398.
- Quiñonero-Candela, J. and Rasmussen, C. E. (2005). A unifying view of sparse approximate gaussian process regression. *The Journal of Machine Learning Research*, 6:1939–1959.
- Rahimi, A. and Recht, B. (2007). Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184.
- Rasmussen, C. and Williams, C. (2006). *Gaussian processes for machine learning*.
- Ruf, T. (1999). The lomb-scargle periodogram in biological rhythm research: analysis of incomplete and unequally spaced time-series. *Biological Rhythm Research*, 30(2):178–201.
- Sakoe, H. and Chiba, S. (1971). A dynamic programming approach to continuous speech recognition. In *Proceedings of the Seventh International Congress on Acoustics*, volume 3, pages 65–69.
- Scargle, J. D. (1982). Studies in astronomical time series analysis. ii-statistical aspects of spectral analysis of unevenly spaced data. *The Astrophysical Journal*, 263:835–853.
- Schulz, M. and Stattegger, K. (1997). Spectrum: Spectral analysis of unevenly spaced paleoclimatic time series. *Computers & Geosciences*, 23(9):929–945.
- Smola, A., Gretton, A., Song, L., and Schölkopf, B. (2007). A hilbert space embedding for distributions. In *Algorithmic Learning Theory*, pages 13–31. Springer.
- Tropp, J. A. (2012a). User-friendly tail bounds for sums of random matrices. *Foundations of Computational Mathematics*, 12(4):389–434.
- Tropp, J. A. (2012b). User-friendly tools for random matrices: An introduction. Technical report, DTIC Document.
- Williams, C. and Seeger, M. (2001). Using the Nyström method to speed up kernel machines. In *Proceedings of the 14th Annual Conference on Neural Information Processing Systems*.
- Xing, E. P., Jordan, M. I., Russell, S., and Ng, A. Y. (2002). Distance metric learning with application to clustering with side-information. In *Advances in neural information processing systems*, pages 505–512.
- Yang, T., Li, Y.-F., Mahdavi, M., Jin, R., and Zhou, Z.-H. (2012). Nyström method vs random fourier features: A theoretical and empirical comparison. In *Advances in neural information processing systems*, pages 476–484.

---

# Complexity of the Exact Solution to the Test Sequencing Problem

---

**Wenhao Liu**

Management Science and Engineering Dept  
Stanford University  
Stanford, California 94305  
owenliu@stanford.edu

**Ross D. Shachter**

Management Science and Engineering Dept  
Stanford University  
Stanford, California 94305  
shachter@stanford.edu

## Abstract

Consider a doctor choosing a treatment for an uncertain disorder for which there are  $n$  costly tests available. Based on the test results observed so far, the doctor can either order another test or proceed to the treatment decision. Although test sequencing is a problem that arises frequently in many decision situations, finding an exact solution is NP-hard with respect to  $n$ . In this paper, we analyze the time complexity of classic symmetric and asymmetric formulations, using influence diagrams and decision trees, to the general test sequencing problem, making no assumptions of conditional independence among the test results. We develop an alternative influence diagram formulation that scales better, and show how a decision circuit formulation improves even more on the decision tree solution through recursive coalescence. We prove that this decision circuit formulation achieves the lower bound complexity for any method for the general test sequencing problem that examines the entire policy space. As a result, the problem is tractable for much larger  $n$  than has been possible to date.

## 1 INTRODUCTION

Consider a doctor who must choose a treatment for an uncertain disorder for which there are  $n$  costly tests available. After each test is performed, the doctor can order another test or proceed to the treatment decision. Therefore, the doctor would like an optimal strategy for sequencing the tests, taking into account the costs of the tests and all of the test results observed so far. In this paper we examine the computational time complexity of exact solution methods for this problem.

The test sequencing problem is an asymmetric decision problem, where many combinations of uncertain variable states given decision variables have zero probability

(Bielza and Shenoy, 1999). The problem arises frequently in many practical decision situations and is of theoretical interest in operations research, machine learning, and the design of experiments. Although finding an exact solution is NP-hard with respect to the number of tests  $n$  (Papadimitriou and Tsitsiklis, 1987), the complexity of standard algorithms for this problem has not been analyzed or compared explicitly in the literature. Traditional graphical models used to solve such problems include decision trees (von Neumann and Morgenstern, 1944) and influence diagrams (Howard and Matheson, 1981; Shachter, 1986). Decision trees with limited coalescence (Howard, 1977; Olmsted, 1983) reuse some of the calculations to improve efficiency. There are many other asymmetric decision model representations, primarily designed to improve model formulation, including asymmetric influence diagrams (Smith et al., 1993), valuation networks (Shenoy, 2000), sequential decision diagrams (Covaliu and Oliver, 1995), sequential valuation networks (Demirer and Shenoy, 2006), and unconstrained influence diagrams (Jensen and Vomlelová, 2002). See Bielza et al. (2011) for a review and comparison of such models.

The test sequencing problem involves a decision maker with one key high-stakes decision and a set of information gathering activities. There has been rich and varied research on this problem dating back to the initial work on dynamic programming (Bellman, 1956), but due to the complexity of the problem much subsequent work in this area has focused on designing heuristic and approximate solution methods. Some exact methods feature state variables representing the belief of the decision maker, instead of maintaining all of the observations in the state, such as in Ulu and Smith (2009). Bickel and Smith (2006) consider a test sequencing problem in the context of oil exploration, and advocate “recombining” decision tree models for efficient computation, an example of the recursive coalescence that we focus on in this paper. Despite all of the research in this area, however, there had been no significant improvements in computational complexity for exact solutions to the general test sequencing problem.



Decision circuits, a generalization of decision trees, were developed by Bhattacharjya and Shachter (2007) and Bhattacharjya (2009) to efficiently evaluate influence diagrams with methods similar to arithmetic circuits for Bayesian networks (Darwiche, 2003). Although decision circuits share similar “tree-like” structures with decision trees, especially decision trees with coalescence, they are also as capable of exploiting conditional independence as influence diagrams, and as able to decompose problems as junction trees (Shachter and Peot, 1992; Jensen et al., 1994).

In the next section we formally define the general test sequencing problem, and in the following sections we formulate and analyze the complexity of exact solutions using the symmetric approach of influence diagrams and the asymmetric approaches of decision trees and decision circuits. We show that the decision circuit achieves the lower bound complexity of any algorithm for the problem that examines the entire policy space. We conclude with a comparison of the results and their implications.

## 2 NOTATION AND FRAMEWORK

In the *general test sequencing problem*, a doctor is treating a patient who has an uncertain disorder  $D$  with  $b$  possible states and probability distribution  $\Pr\{D\}$ . The doctor will choose a treatment  $T_x$  from among  $a$  alternatives in order to maximize the expected dollar value  $V$  from the treatment. Before making the treatment choice, there are  $n$  possible tests available, each with at most  $c$  possible results. Although we will evaluate the costs of the tests in dollars, the costs could also arise in practice from the delay in treatment or the side effects from performing the tests. At any point in time the doctor can make the treatment decision or order another test, knowing the results of all of the tests that have been previously ordered. Because the choice the doctor makes can depend on the results of those tests, the policy space of the test sequencing problem is exponential in the number of tests  $n$ ,  $(c + 1)^n$ , the number of possible sets of observations available in the decision making process.

Each test  $T_m$  has an associated cost  $C_m > 0$ ,  $m = 1, \dots, n$ , and we assume that the time horizon is short enough that its cost does not depend on when that particular test was performed. Therefore, the expected value of the prospect of any scenario is the difference between the dollar value of the treatment,  $E[V|T_x, T_1, \dots, T_n]$ , and the costs of the tests ordered,  $C_1 + \dots + C_n$ . Such a value model is said to be *separable*, and we exploit that in our formulations. We assume, without loss of generality, that the same test will produce the same result if performed more than once, so it would never be optimal to order the same test twice. (A test that might be worth repeating could be included as multiple available tests.) The test sequencing

problem is *general* as we impose no (conditional) independence assumptions on the  $n$ -vector of observable test results  $\mathbf{R}$  conditioned on the disorder  $D$ , with probability distribution  $\Pr\{\mathbf{R}|D\}$ .

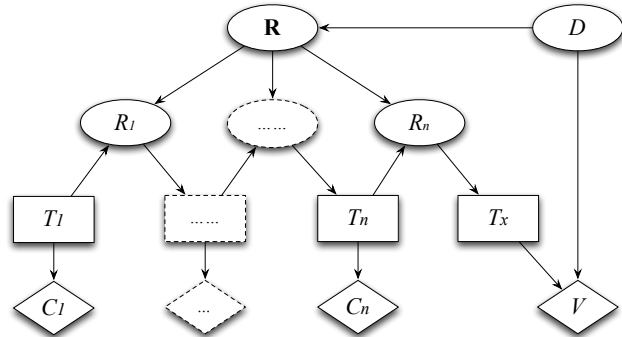


Figure 1: Influence diagram for the general test sequencing problem with  $n$  tests

An influence diagram for the general test sequencing problem is shown in Figure 1. There are  $n$  testing decisions,  $T_1, \dots, T_n$ , each with  $n + 1$  alternatives (including not to test), and the treatment decision,  $T_x$ . Corresponding to each of the testing decisions  $T_m$  there is a dollar cost  $C_m$  of testing, depending on the test chosen, and an expected dollar value  $V$  that depends on both the disorder  $D$  and the treatment decision  $T_x$ ,  $E[V|D, T_x]$ . The potentially observable test results,  $\mathbf{R}$ , depends on  $D$ , and the actual test result  $R_m$  observed after testing decision  $T_m$  is therefore a deterministic function of  $T_m$  and  $\mathbf{R}$ ,

$$R_m = \begin{cases} \mathbf{R}_{T_m} & \text{if } T_m \neq 0 \\ 1 & \text{if } T_m = 0 \end{cases} \quad \text{for } m = 1, \dots, n. \quad (1)$$

Earlier observations and decisions are known at the time of later decisions (Howard, 1977). Therefore, at the time of decision  $T_m$  the doctor will know which tests were ordered,  $T_1, \dots, T_{m-1}$ , and their corresponding results,  $R_1, \dots, R_{m-1}$ . At the time of the treatment decision  $T_x$  the doctor will know all tests that were ordered,  $T_1, \dots, T_n$ , and their results,  $R_1, \dots, R_n$ . These definitions are summarized in Table 1. In the following sections we will consider different approaches to the exact solution of this problem.

## 3 INFLUENCE DIAGRAM SOLUTIONS

In this section we formulate an influence diagram solution to the general test sequencing problem and a more efficient formulation based on a Markov Decision Process (MDP) model. This method is symmetric in the sense that the probability distributions are full arrays and all variables are included in the formulations of every scenario.

| Symbol         | Definition  |
|----------------|---|
| $T_x$          | treatment decision  |
| $a$            | number of treatment alternatives                                  |
| $D$            | uncertain disorder  |
| $b$            | number of disorder states   |
| $V$            | expected dollar value of the treatment $T_x$ for the disorder $D$ |
| $n$            | number of tests available   |
| $\mathbf{R}$   | $n$ -vector of potentially observable test results                |
| $c$            | maximum number of possible results for each test                  |
| $T_m$          | decision which test to order $m$ th, $m = 1, \dots, n$            |
| $R_m$          | results of the $m$ th test ordered                                |
| $C_m$          | cost of the $m$ th test ordered                                   |
| $\mathbf{S}_m$ | $n$ -vector of test results observed after $m$ decisions          |

Table 1: Symbol Definitions

We can solve the influence diagram shown in Figure 1 by constructing a rooted cluster tree, as shown in Figure 2 (Shachter and Peot, 1992), similar for our purposes to a strong junction tree (Jensen et al., 1994), and minimal because these cliques are necessary to represent the problem (Shachter, 1999). The computational time complexity of the solution is determined by the total of the sizes of cluster tables,  $abc^n + ac^{2n}(n+1)^n$ , recognizing that the number of possible states for  $T_x$ ,  $D$ ,  $\mathbf{R}$ ,  $T_m$ , and  $R_m$  are  $a$ ,  $b$ ,  $c^n$ ,  $n+1$ , and  $c$ , respectively.

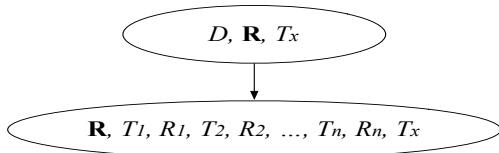


Figure 2: Rooted cluster tree for the influence diagram in Figure 1

**Theorem 1.** *The computational complexity of the standard influence diagram formulation of the general test sequencing problem is  $O(c^{2n}(n+1)^n)$ .*

An influence diagram formulation based on an MDP model, however, is more efficient than the standard influence diagram model for large  $n$ . The key is to introduce a *Markov state variable*,  $\mathbf{S}_m$ , the observed test results after  $m$  testing decisions, which renders past observations and decisions independent of future decisions.  $\mathbf{S}_m$  is an  $n$ -vector, with components corresponding to the different possible test results, but with  $c+1$  possible values for

each component, including a new state “0” corresponding to the “test results not yet observed”. Therefore, letting  $\mathbf{S}_0 = \mathbf{0}$  indicate that no tests have been performed before the first testing decision, we can define  $\mathbf{S}_m$  for each possible  $j, m = 1, \dots, n$  as a deterministic function of  $\mathbf{S}_{m-1}$ ,  $T_m$ , and  $\mathbf{R}$  by

$$(\mathbf{S}_m)_j = \begin{cases} \mathbf{R}_{T_m} & \text{if } T_m = j \neq 0 \\ (\mathbf{S}_{m-1})_j & \text{otherwise} \end{cases} \quad (2)$$

With this definition of  $\mathbf{S}_m$  we can formulate the influence diagram shown in Figure 3. Because of the Markov state, the decisions  $T_2, \dots, T_n$  and  $T_x$  depend only on the corresponding state variables  $\mathbf{S}_1, \dots, \mathbf{S}_n$ , respectively, rather than any of the past decisions and observations. However, this is not yet an MDP and would not be efficient to solve because of the role played by the uncertain potentially observable test results  $\mathbf{R}$ . Nevertheless, the definition of  $\mathbf{S}_m$  allows us to reformulate this influence diagram into an MDP influence diagram without  $\mathbf{R}$ , as shown in Figure 4, that is efficient to solve.

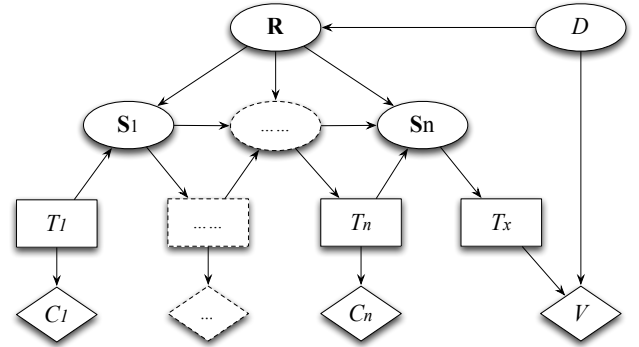


Figure 3: The influence diagram of the general test sequencing problem with Markov states

**Theorem 2.** *The influence diagram shown in Figure 4 is a valid representation of the general test sequencing problem.*

*Proof.* Given the relationships represented by the influence diagram shown in Figure 3,  $D$  is conditionally independent of  $T_1, \dots, T_n, \mathbf{S}_1, \dots, \mathbf{S}_n$  given  $\mathbf{R}$ . By the definition of  $\mathbf{S}_m$ ,  $\mathbf{R}$  is conditionally independent of  $T_1, \dots, T_n, \mathbf{S}_1, \dots, \mathbf{S}_{n-1}$  given  $\mathbf{S}_n$ . Therefore, it follows that  $D$  must be conditionally independent of  $T_1, \dots, T_n, \mathbf{S}_1, \dots, \mathbf{S}_{n-1}$  given  $\mathbf{S}_n$ . Likewise, by the definition of  $\mathbf{S}_m$ ,  $\mathbf{S}_{m+1}$  is conditionally independent of  $T_1, \dots, T_m, \mathbf{S}_1, \dots, \mathbf{S}_{m-1}$  given  $\mathbf{S}_m$  and  $T_{m+1}$  for  $m = 1, \dots, n-1$ , as shown in Figure 4.  $\square$

Even though the earlier testing decisions and their results will be known at the time of later decisions, it is sufficient to observe the Markov state  $\mathbf{S}_m$  as shown in the MDP

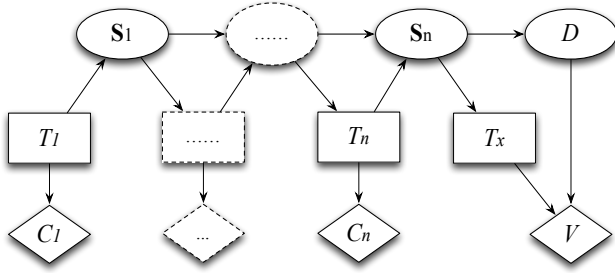


Figure 4: The MDP influence diagram for the general test sequencing problem

influence diagram (Figure 4). Therefore, we can solve it using the rooted cluster tree shown in Figure 5. As before, the computational time complexity of the solution is determined by the total of the sizes of cluster tables,  $(n + 1)(c + 1)^n + (n - 1)(n + 1)(c + 1)^{2n} + ab(c + 1)^n$ , recognizing that the number of possible states for  $T_x$ ,  $D$ ,  $T_m$ , and  $S_m$  are  $a, b, n + 1$ , and  $(c + 1)^n$ , respectively. We must also account for the cost of preprocessing to reformulate the diagram to the MDP at a complexity of  $bc^n(c + 1)^n + (n + 1)(c + 1)^{2n}$ . Although this is dominated by the expression above, we include it in our final comparisons.

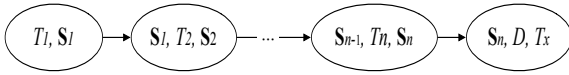


Figure 5: The rooted cluster tree for the MDP influence diagram

**Theorem 3.** *The computational complexity of the MDP influence diagram formulation of the general test sequencing problem is  $O(n^2(c + 1)^{2n})$ .*

#### 4 PURE DECISION TREE SOLUTION

In this section we formulate and analyze the computational complexity of the exact solution to the general test sequencing problem using an asymmetric pure decision analysis decision tree without *coalescence*, the reuse of sub-tree calculations. Such decision trees maintain the strict tree structure in which each node has at most one parent. Despite the prevalence and usefulness of decision tree models, there has been limited evaluation of their complexity when applied to asymmetric decision problems.

A decision tree is a natural representation for the asymmetry in the general test sequencing problem, recognizing that after we have observed  $m$  test results, there are only  $n - m$  remaining tests to consider and the choice of which test to order, if any, can depend on the test results that we have already observed. To build a decision tree we will

need to preprocess the probability distributions for  $D$ ,  $\mathbf{R}$ , and  $R_1, \dots, R_n$  from the assessed distributions as shown in the influence diagram in Figure 1 to the inferential order they need to appear in the decision tree, where  $D$  and  $\mathbf{R}$  are not observed before any of the decisions. The computational effort to perform this pre-processing is substantial,  $O(bc(c + 1)^n)$ , but dominated by the work needed to evaluate the decision tree. Hence, we can ignore it in our analysis but include it in our final comparisons.

For each possible sequence of  $m$  tests and their observed test results,  $m = 0, \dots, n$ , there is a decision node in the decision tree corresponding to the choice of treatment or of another test. There are  $\binom{n}{m}m!c^m$  such possible sequences in the tree.

**Proposition 1.** *The total number of decision nodes in a pure decision tree without coalescence is*

$$\sum_{m=0}^n \binom{n}{m} m! c^m.$$

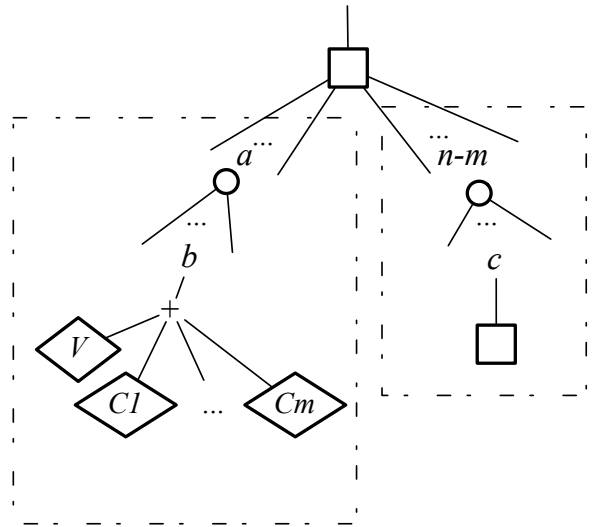


Figure 6: A generic decision node in the pure decision tree without coalescence, with  $m$  observed test results

The computational time complexity of the decision tree solution is determined by the number of arcs (or nodes) in the tree. Figure 6 shows a generic decision node within the decision tree whose ancestors include exactly  $m$  tests and their corresponding test results. There are  $a + n - m$  alternatives, corresponding to choosing from one of the remaining  $n - m$  tests or choosing to stop testing and make the treatment decision. For each of the test alternatives there are  $c$  possible test results, each leading to a different decision node in the tree, and for each of the  $a$  treatment alternatives there are  $b$  possible disorder states. Therefore, for each of the decision nodes in the tree there are

$a + ab + (n - m)c$  arcs in the tree, and the total number of arcs is given by

$$\begin{aligned}
& \sum_{m=0}^n \binom{n}{m} m! c^m [(a + ab) + (n - m)c] \\
= & (a + ab) \sum_{m=0}^n \frac{n!}{(n - m)!} c^m + \sum_{m=0}^n \frac{n!}{(n - m - 1)!} c^{m+1} \\
= & (a + ab) c^n n! \sum_{m=0}^n \frac{c^{-m}}{m!} + c^n n! \sum_{m=0}^{n-1} \frac{c^{-m}}{m!} \\
\approx & (a + ab) c^n n! e^{1/c} + c^n n! e^{1/c} \\
= & (a + ab + 1) c^n n! e^{1/c} = O(c^n n!)
\end{aligned}$$

**Theorem 4.** *The computational complexity of the decision tree formulation with no coalescence of the general test sequencing problem is  $O(c^n n!)$ .*

We will see in the next section that we can improve on the efficiency of the influence diagram and decision tree by allowing recursive coalescence, reusing subtree calculations as much as possible. Coalescence has traditionally been applied with decision trees in a limited fashion, at most once for any path in the tree (Howard, 1977; Olmsted, 1983), but we will apply it much more extensively in a generalization of decision trees called decision circuits.

## 5 DECISION CIRCUIT SOLUTION

In this section we formulate a decision circuit solution to the general test sequencing problem and show that it achieves the lower bound complexity of any algorithm for the problem that examines the entire policy space. Although this solution can be viewed as an extension of the pure decision tree solution with recursive coalescence, the decision circuit naturally integrates such coalescence and, unlike decision trees, does not need the distributions to be preprocessed.

### 5.1 INTRODUCTION TO DECISION CIRCUITS

Decision circuits are generalized decision trees that maintain their asymmetry while exploiting any conditional independence. They were developed by Bhattacharjya and Shachter (2007) and Bhattacharjya (2009) to efficiently evaluate influence diagrams with methods similar to arithmetic circuits for Bayesian networks (Darwiche, 2003). A decision problem represented by an influence diagram, or an intermediate structure, a decision circuit backbone, can be transformed into a decision circuit for efficient evaluation and sensitivity analysis (Shachter and Bhattacharjya, 2010; Bhattacharjya and Shachter, 2008, 2010). Although decision circuits were not developed as a repre-

sentation for communication, Bhattacharjya and Shachter (2012) showed how formulating asymmetric decision circuits directly, instead of formulating an influence diagram and transforming it into a decision circuit, could be desirable in many applications. They also showed how to build decision circuits in assessed form, avoiding the probability distribution preprocessing effort (Bayes Theorem “tree flipping”) needed for decision trees.

Decision circuits generalize decision trees in several key ways. Both decision circuits and decision trees are natural representations for asymmetric problems. Extensive, even recursive, coalescence is encouraged in the decision circuit, by allowing nodes to have multiple parents. Because the expectation operation for an uncertain variable in decision trees is represented as separate *sum* and *product* operations in the decision circuit, the probability distribution corresponding to the variable can appear further downstream (Shenoy, 1998). As a result, probability distributions can be incorporated into the decision circuit as assessed and there is no need for preprocessing the assessed distributions. Decision circuits are also able to exploit separable problem structure as found in influence diagrams and junction trees (Tatman and Shachter, 1990; Shachter and Peot, 1992; Jensen et al., 1994). Finally, once the probability and value distributions are specified the decision circuit can be compiled for even greater efficiency.

### 5.2 DECISION CIRCUIT FORMULATION

An example of our decision circuit formulation is shown in Figure 7 for the general test sequencing problem with  $a = b = c = n = 2$ , that is, there are two treatment alternatives, two disorder states, and two tests available with two test results each. Decision circuits can have “indicator variables”  $\lambda$  to control and manage evidence and sensitivity analysis, but we have omitted them to simplify the diagram. Including the indicators would not significantly affect the computational complexity for the test sequencing problem.

At the leaves of the decision circuit shown in Figure 7 are the expected dollar values for each prospect  $E[V|D, T_x]$ , the costs  $C$  for each type of test, the probabilities  $\Pr\{D\}$ , and the likelihoods  $\Pr\{\mathbf{R}|D\}$ . The separable costs are incorporated using *branching sum* (“B+”) nodes (Shachter and Bhattacharjya, 2010). The quantities at the leaves are combined and reused throughout the circuit both as probabilities and as unnormalized probability-value hybrids for decision making. For example, we can marginalize for the case where some of the tests are not performed. The decision circuit is evaluated by sweeping up from the leaves to the root and, in the process, making all of the decisions at *max* nodes and determining the optimal expected value (Darwiche, 2003; Bhattacharjya and Shachter, 2007). A sweep down through the circuit computes derivatives of the optimal value with respect to any

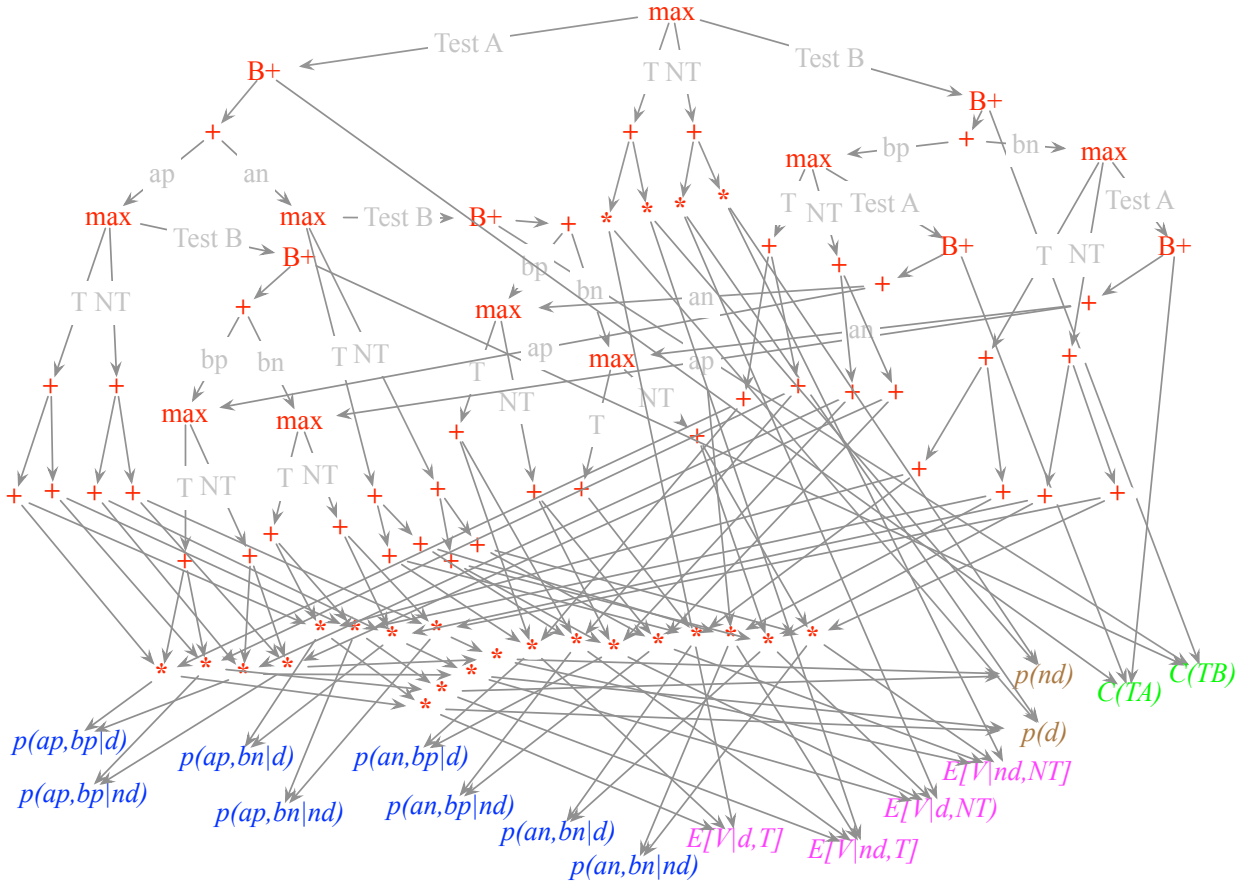


Figure 7: The decision circuit for general test sequencing involving two treatment alternatives, two disorder states, and two tests with two test results each

of the nodes and assessed parameters for use in sensitivity analysis (Bhattacharjya and Shachter, 2008, 2010). Therefore, we can compute the computational time complexity of a decision circuit by counting the number of arcs.

The coalescence in the decision circuit allows us to exploit essential properties of the general test sequencing problem. For example, when the results from  $m$  tests have been observed, the order those tests were performed does not matter (Jaynes, 2003; Bickel and Smith, 2006). In our decision circuit those  $m!$  different test sequences (corresponding to  $m$  equivalent observed test results) all lead to the same decision ( $max$ ) node for the selection of an  $(m+1)$ st test or to make the treatment decision. The structure of our circuit relies on the fact that the same choice will be optimal regardless what order the observed tests were performed, similar to the Markov state in our MDP formulation or an “information set” in game theory (Shenoy, 1998).

### 5.3 THE COMPLEXITY OF THE DECISION CIRCUIT FORMULATION

In our decision circuit formulation, for each possible set of  $m$  tests and their observed test results,  $m = 0, \dots, n$ , there is a decision node in the decision tree corresponding to the choice of treatment or another test. There are  $\binom{n}{m} c^m$  such possible sets in the circuit. Note that, by contrast, in the decision tree there was a distinction about the order of the tests, increasing the number of decision nodes by a factor of  $m!$ . The binomial theorem provides a closed-form expression for the number of decision nodes in the decision circuit.

**Proposition 2.** *The total number of decision nodes in the decision circuit is*

$$\sum_{m=0}^n \binom{n}{m} c^m = (c+1)^n.$$

The number of arcs in the decision circuit determines the computational time complexity for the circuit. Figure 8 shows a generic decision node within the decision circuit whose ancestors include exactly  $m$  tests and their cor-

responding test results. There are  $m$  arcs into the node, corresponding to the  $m$  different tests that could have been observed last. There are  $a+n-m$  alternatives, corresponding to choosing from one of the remaining  $n-m$  tests or choosing to stop testing and make the treatment decision. For each of the  $a$  treatment alternatives and  $b$  possible disorder states we marginalize over  $c$  possible test results (except when  $m$  is either 0 or  $n$ ) to compute the probabilities and probability-value hybrids of the disease and observed test results from those used by decisions with  $m+1$  test results. For each of the  $n-m$  test alternatives there are three arcs into or out of the branching sum node used to register the separable cost of the test, followed by  $c$  test results, each leading to a different decision node in the circuit, and counted as incoming arcs for those nodes. Therefore, for each of the decision nodes in the circuit there are less than  $m+(a+ab+abc)+3(n-m)$  arcs in the decision circuit. We will compute the complexity from each of these three terms separately.

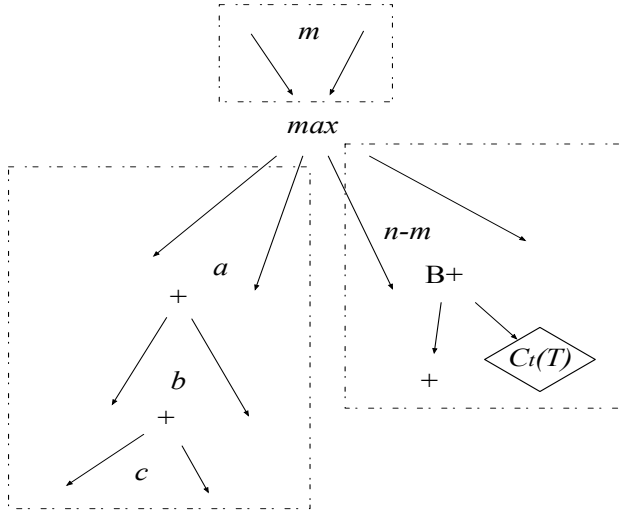


Figure 8: A generic decision node in the decision circuit with  $m$  observed test results

The first term corresponds to the  $m$  arcs directed into the decision node. The total for all decision nodes is

$$\begin{aligned}
 \sum_{m=0}^n \binom{n}{m} m c^m &= \sum_{m=1}^n \binom{n}{m} m c^m \\
 &= \sum_{m=1}^n \frac{n!}{(m-1)!(n-m)!} c^m \\
 &= n c \sum_{m=1}^n \frac{(n-1)!}{(m-1)!(n-m)!} c^{m-1} \\
 &= n c (c+1)^{n-1}.
 \end{aligned} \tag{3}$$

The second term is the less than  $abc$  arcs corresponding to the treatment alternatives. The total for all decision

nodes is less than

$$(a+ab+abc) \sum_{m=0}^n \binom{n}{m} c^m = (a+ab+abc)(c+1)^n. \tag{4}$$

The third term is the  $3(n-m)$  arcs corresponding to the next tests and the arcs needed to register the cost of the test. The total for all decision nodes is

$$\begin{aligned}
 3 \sum_{m=0}^{n-1} \binom{n}{m} (n-m) c^m &= 3 \sum_{m=0}^{n-1} \frac{n!}{m!(n-m)!} (n-m) c^m \\
 &= 3n \sum_{m=0}^{n-1} \frac{(n-1)!}{m!(n-1-m)!} c^m \\
 &= 3n(c+1)^{n-1}.
 \end{aligned} \tag{5}$$

Finally, including the  $2ab(c^n+1)$  arcs at the bottom of the decision circuit that incorporate the  $E[V|D, T_x]$ ,  $\Pr\{D\}$ , and  $\Pr\{\mathbf{R}|D\}$  tables, the total number of arcs is less than

$$2ab(c^n+1) + n(c+3)(c+1)^{n-1} + (a+ab+abc)(c+1)^n,$$

and we have shown the following result.

**Theorem 5.** *The computational complexity of the decision circuit formulation of the general test sequencing problem is  $O(n(c+1)^{n-1})$ .*

In general the test sequencing problem can have an arbitrary optimal policy for any of the  $(c+1)^n$  possible sets of observations. Given  $m$  tests have been performed and any of the  $c^m$  possible test results, the corresponding policy is determined by comparing the net expected values among the remaining  $n-m$  tests that could be performed. The number of such comparisons is  $\sum_{m=0}^{n-1} \binom{n}{m} (n-m) c^m$ , similar to Equation 5, and this provides a lower bound on the complexity of any algorithm for the general problem that examines the entire policy space,  $O(n(c+1)^{n-1})$ . Because the decision circuit formulation achieves this bound, the bound must be tight.

**Theorem 6.** *Any algorithm for the general test sequencing problem that examines the entire policy space of test results must have time complexity with lower bound  $\Omega(n(c+1)^{n-1})$ , as achieved by the decision circuit formulation.*

We compare the computational complexity of the four models with the parameters  $a=b=c$  set to 2 and 3. The results as a function of the number of tests available,  $n$ , are displayed in Table 2, and the logs of the complexities are shown in Figure 9 for  $a=b=c=2$ . The plots would appear similar, but with greater slopes, if we increased the values of the parameters  $a$ ,  $b$ , and/or  $c$ .

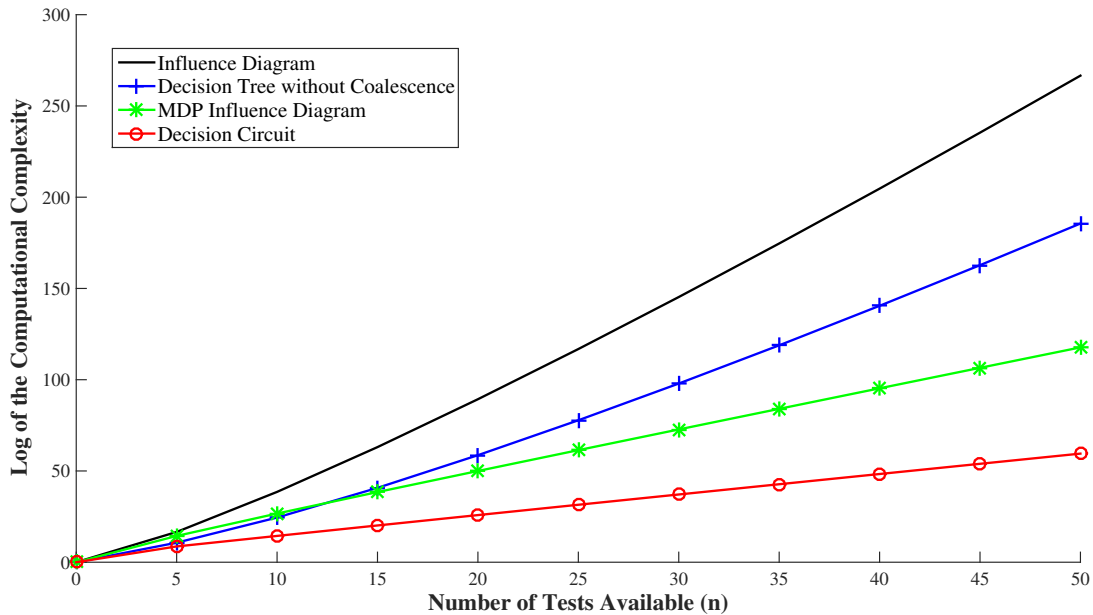


Figure 9: Log of the computational complexity of different formulations with model parameters  $a = b = c = 2$

| State Space     | Formulation                       | 5 Tests            | 10 Tests              | 15 Tests              | 20 Tests              | 25 Tests              |
|-----------------|-----------------------------------|--------------------|-----------------------|-----------------------|-----------------------|-----------------------|
| $a = b = c = 2$ | Decision Tree without Coalescence | $4.43 \times 10^4$ | $4.29 \times 10^{10}$ | $4.95 \times 10^{17}$ | $2.94 \times 10^{25}$ | $6.00 \times 10^{33}$ |
|                 | Standard Influence Diagram        | $1.59 \times 10^7$ | $5.44 \times 10^{16}$ | $2.48 \times 10^{27}$ | $6.12 \times 10^{38}$ | $5.33 \times 10^{50}$ |
|                 | MDP Influence Diagram             | $1.79 \times 10^6$ | $3.84 \times 10^{11}$ | $4.94 \times 10^{16}$ | $5.11 \times 10^{21}$ | $4.67 \times 10^{26}$ |
|                 | Decision Circuit                  | $5.69 \times 10^3$ | $1.82 \times 10^6$    | $5.60 \times 10^8$    | $1.65 \times 10^{11}$ | $4.72 \times 10^{13}$ |
| $a = b = c = 3$ | Decision Tree without Coalescence | $5.29 \times 10^5$ | $3.89 \times 10^{12}$ | $3.40 \times 10^{20}$ | $1.54 \times 10^{29}$ | $2.38 \times 10^{38}$ |
|                 | Standard Influence Diagram        | $1.38 \times 10^9$ | $2.71 \times 10^{20}$ | $7.12 \times 10^{32}$ | $1.01 \times 10^{46}$ | $5.10 \times 10^{59}$ |
|                 | MDP Influence Diagram             | $3.22 \times 10^7$ | $1.21 \times 10^{14}$ | $2.77 \times 10^{20}$ | $5.08 \times 10^{26}$ | $8.24 \times 10^{32}$ |
|                 | Decision Circuit                  | $5.20 \times 10^4$ | $5.77 \times 10^7$    | $6.63 \times 10^{10}$ | $7.59 \times 10^{13}$ | $8.61 \times 10^{16}$ |

Table 2: Computational complexity of different formulations with model parameters  $a = b = c$

## 6 CONCLUSIONS

The general test sequencing problem has been known to be NP-hard with respect to the number of tests available, so most past research efforts have searched for approximate solution methods, heuristics, simulations, and simplifying assumptions.

In this paper, we develop a decision circuit formulation for the general test sequencing problem that solves the problem exactly without imposing any additional assumptions. We analyze its computational complexity and compare it with other frequently used sequential decision making models, pure decision trees and influence diagrams, and we develop a more efficient influence diagram model based on an MDP. In our comparison, the decision circuit model significantly outperforms the others. In fact, it achieves

the lower bound on the computational complexity for any method for the general test sequencing problem that examines the entire policy space.

We could have obtained this same order of computational complexity by implementing decision trees with recursive coalescence. To construct such decision trees would have required explicit preprocessing of the assessed distributions, which is more efficiently done implicitly within our decision circuit formulation. It would also have used coalescence in a recursive manner natural in decision circuits but not common for decision trees.

In a similar fashion, we could have obtained a more efficient MDP formulation by recognizing the structural asymmetry in the problem. Because we observe at most one test result in each time period we could prune from

the state space those paths with more observations. This would improve the run-time computational complexity of the MDP and lead to a problem structure quite similar to the decision circuit.

Another advantage of using decision circuits is the efficient sensitivity analysis available on the assessed problem parameters, the probabilities and costs. The decision circuit formulation developed here allows us to solve the general test sequencing problem exactly for much larger values of  $n$  than has been possible before.

Information gathering decisions are strategic components in many decision problems in medicine, engineering, and other domains where they can significantly improve performance (Bickel and Smith, 2006). This paper has focused on the general test sequencing problem, with no assumptions, such as probabilistic independence. We realize that this is a specific class of problems, but we have used its structure to analyze and compare different exact techniques, and to demonstrate that a relatively new approach, decision circuits, allows us to achieve the complexity lower bound. This suggests that well-crafted decision circuits might perform relatively well on more general problems, such as troubleshooting, where observations and actions are interspersed and actions affect both the state of the system and future observations (Breese and Heckerman, 1996).

### Acknowledgements

We thank the anonymous referees for their suggestions and Yuval Shahar for inspiring the research.

### References

- Richard Bellman. A problem in the sequential design of experiments. *Sankhyā: The Indian Journal of Statistics (1933-1960)*, 16(3/4):221–229, 1956.
- Debarun Bhattacharjya. *Decision circuits: A graphical representation for efficient decision analysis computation*. PhD thesis, Stanford University, 2009.
- Debarun Bhattacharjya and Ross D Shachter. Evaluating influence diagrams with decision circuits. In *Proceedings of the Twenty-Third Annual Conference on Uncertainty in Artificial Intelligence (UAI-07)*, pages 9–16, 2007.
- Debarun Bhattacharjya and Ross D Shachter. Sensitivity analysis in decision circuits. In *Proceedings of the Twenty-Fourth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-08)*, pages 34–42, 2008.
- Debarun Bhattacharjya and Ross D Shachter. Three new sensitivity analysis methods for influence diagrams. In *Proceedings of the Twenty-Sixth Annual Conference on Uncertainty in Artificial Intelligence (UAI-10)*, pages 56–64, 2010.
- Debarun Bhattacharjya and Ross D Shachter. Formulating Asymmetric Decision Problems as Decision Circuits. *Decision Analysis*, 9(2):138–145, 2012.
- Eric J Bickel and James E Smith. Optimal sequential exploration: A binary learning model. *Decision Analysis*, 3(1):16–32, 2006.
- Concha Bielza and Prakash P Shenoy. A Comparison of Graphical Techniques for Asymmetric Decision Problems. *Management Science*, 45(11):1552–1569, 1999.
- Concha Bielza, Manuel Gómez, and Prakash P Shenoy. A review of representation issues and modeling challenges with influence diagrams. *Omega*, 39(3):227–241, 2011.
- John S Breese and David Heckerman. Decision-theoretic troubleshooting: A framework for repair and experiment. In *Proceedings of the Twelfth Annual Conference on Uncertainty in Artificial Intelligence (UAI-96)*, 1996.
- Zvi Covaliu and Robert M. Oliver. Representation and solution of decision problems using sequential decision diagrams. *Management Science*, 41(12):pp. 1860–1881, 1995.
- Adnan Darwiche. A differential approach to inference in Bayesian networks. *J. ACM*, 50(3):280–305, 2003.
- Riza Demirer and Prakash P Shenoy. Sequential valuation networks for asymmetric decision problems. *European Journal of Operational Research*, 169(1):286–309, 2006.
- Ronald A Howard. The used car buyer. In *Readings in Decision Analysis*. Stanford Research Institute, Menlo Park, CA, 1977.
- Ronald A Howard and James E Matheson. Influence Diagrams. *R. A. Howard, J. E. Matheson, eds. The Principles and Applications of Decision Analysis*. 1984, 2: 719–762, 1981.
- Edwin T Jaynes. *Probability Theory: The Logic of Science*. Cambridge University Press, 2003.
- Finn V Jensen and Marta Vomlelová. Unconstrained influence diagrams. In *Proceedings of the Eighteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-02)*, pages 234–241, 2002.
- Frank Jensen, Finn V Jensen, and Søren L Dittmer. From influence diagrams to junction trees. In *Proceedings of the Tenth international conference on Uncertainty in artificial intelligence*, pages 367–373. Morgan Kaufmann Publishers Inc., 1994.
- Scott M Olmsted. *On Representing and Solving Decision Problems*. PhD thesis, Stanford University, 1983.
- Christos H Papadimitriou and John N Tsitsiklis. The Complexity of Markov Decision Processes. *Mathematics of Operations Research*, 12:441–450, 1987.



- Ross D Shachter. Evaluating influence diagrams. *Operations Research*, 34(6):871–882, 1986.
- Ross D. Shachter. Efficient value of information computation. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI-99)*, pages 594–601, 1999.
- Ross D Shachter and Debarun Bhattacharjya. Dynamic programming in influence diagrams with decision circuits. In *Proceedings of the Twenty-Sixth Annual Conference on Uncertainty in Artificial Intelligence (UAI-10)*, pages 509–516, 2010.
- Ross D. Shachter and Mark A. Peot. Decision making using probabilistic inference methods. In *Proceedings of the Eighth Conference on Uncertainty in Artificial Intelligence (UAI-92)*, pages 276–283, 1992.
- Prakash P Shenoy. Game Trees For Decision Analysis. *Theory and decision*, 44(2):149–171, 1998.
- Prakash P Shenoy. Valuation network representation and solution of asymmetric decision problems. *European Journal of Operational Research*, 121(3):579–608, 2000.
- James E Smith, Samuel Holtzman, and James E Matheson. Structuring conditional relationships in influence diagrams. *Operations Research*, 41(2):280–297, 1993.
- Joseph A Tatman and Ross D Shachter. Dynamic programming and influence diagrams. *Systems, Man and Cybernetics, IEEE Transactions on*, 20(2):365–379, 1990.
- Canan Ulu and James E Smith. Uncertainty, information acquisition, and technology adoption. *Operations Research*, 57(3):740–752, 2009.
- John von Neumann and Oskar Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, Princeton, NJ, 1944.

---

# Finite-Sample Analysis of Proximal Gradient TD Algorithms

---

**Bo Liu**                      **Ji Liu**                      **Mohammad Ghavamzadeh**   **Sridhar Mahadevan**                      **Marek Petrik**  
UMass Amherst              University of Rochester      Adobe & INRIA Lille              UMass Amherst                      IBM Research  
boliu@cs.umass.edu              jliu@cs.rochester.edu      Mohammad.ghavamzadeh@inria.fr      mahadeva@cs.umass.edu              marekpetrik@gmail.com

## Abstract

In this paper, we show for the first time how gradient TD (GTD) reinforcement learning methods can be formally derived as true stochastic gradient algorithms, not with respect to their original objective functions as previously attempted, but rather using derived primal-dual saddle-point objective functions. We then conduct a saddle-point error analysis to obtain finite-sample bounds on their performance. Previous analyses of this class of algorithms use stochastic approximation techniques to prove asymptotic convergence, and no finite-sample analysis had been attempted. Two novel GTD algorithms are also proposed, namely projected GTD2 and GTD2-MP, which use proximal “mirror maps” to yield improved convergence guarantees and acceleration, respectively. The results of our theoretical analysis imply that the GTD family of algorithms are comparable and may indeed be preferred over existing least squares TD methods for off-policy learning, due to their linear complexity. We provide experimental results showing the improved performance of our accelerated gradient TD methods.

## 1 INTRODUCTION

Obtaining a true stochastic gradient temporal difference method has been a longstanding goal of reinforcement learning (RL) [Bertsekas and Tsitsiklis, 1996; Sutton and Barto, 1998], ever since it was discovered that the original TD method was unstable in many off-policy scenarios where the target behavior being learned and the exploratory behavior producing samples differ. Sutton *et al.* [2008, 2009] proposed the family of gradient-based temporal difference (GTD) algorithms which offer several interesting properties. A key property of this class of GTD algorithms is that they are asymptotically off-policy convergent, which was shown using stochastic approximation

[Borkar, 2008]. This is quite important when we notice that many RL algorithms, especially those that are based on stochastic approximation, such as  $TD(\lambda)$ , do not have convergence guarantees in the off-policy setting. Unfortunately, this class of GTD algorithms are *not true stochastic gradient methods with respect to their original objective functions*, as pointed out in Szepesvári [2010]. The reason is not surprising: the gradient of the objective functions used involve products of terms, which cannot be sampled directly, and was decomposed by a rather ad-hoc splitting of terms. In this paper, we take a major step forward in resolving this problem by showing a principled way of designing true stochastic gradient TD algorithms by using a primal-dual saddle point objective function, derived from the original objective functions, coupled with the principled use of *operator splitting* [Bauschke and Combettes, 2011].

Since in real-world applications of RL, we have access to only a finite amount of data, finite-sample analysis of gradient TD algorithms is essential as it clearly shows the effect of the number of samples (and the parameters that play a role in the sampling budget of the algorithm) in their final performance. However, most of the work on finite-sample analysis in RL has been focused on batch RL (or approximate dynamic programming) algorithms (e.g., Kakade and Langford 2002; Munos and Szepesvári 2008; Antos *et al.* 2008; Lazaric *et al.* 2010a), especially those that are least squares TD (LSTD)-based (e.g., Lazaric *et al.* 2010b; Ghavamzadeh *et al.* 2010, 2011; Lazaric *et al.* 2012), and more importantly restricted to the on-policy setting. In this paper, we provide the finite-sample analysis of the GTD family of algorithms, a relatively novel class of gradient-based TD methods that are guaranteed to converge even in the off-policy setting, and for which, to the best of our knowledge, no finite-sample analysis has been reported. This analysis is challenging because **1**) the stochastic approximation methods that have been used to prove the asymptotic convergence of these algorithms do not address convergence rate analysis; **2**) as we explain in detail in Section 2.1, the techniques used for the analysis of the stochastic gradient methods cannot be applied here;

3) finally, the difficulty of finite-sample analysis in the off-policy setting.

The major contributions of this paper include the first finite-sample analysis of the class of gradient TD algorithms, as well as the design and analysis of several improved GTD methods that result from our novel approach of formulating gradient TD methods as true stochastic gradient algorithms w.r.t. a saddle-point objective function. We then use the techniques applied in the analysis of the stochastic gradient methods to propose a unified finite-sample analysis for the previously proposed as well as our novel gradient TD algorithms. Finally, given the results of our analysis, we study the GTD class of algorithms from several different perspectives, including acceleration in convergence, learning with biased importance sampling factors, etc.

## 2 PRELIMINARIES

Reinforcement Learning (RL) [Bertsekas and Tsitsiklis, 1996; Sutton and Barto, 1998] is a class of learning problems in which an agent interacts with an unfamiliar, dynamic and stochastic environment, where the agent’s goal is to optimize some measure of its long-term performance. This interaction is conventionally modeled as a Markov decision process (MDP). A MDP is defined as the tuple  $(\mathcal{S}, \mathcal{A}, P_{ss'}^a, R, \gamma)$ , where  $\mathcal{S}$  and  $\mathcal{A}$  are the sets of states and actions, the transition kernel  $P_{ss'}^a$  specifying the probability of transition from state  $s \in \mathcal{S}$  to state  $s' \in \mathcal{S}$  by taking action  $a \in \mathcal{A}$ ,  $R(s, a) : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$  is the reward function bounded by  $R_{\max}$ , and  $0 \leq \gamma < 1$  is a discount factor. A stationary policy  $\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$  is a probabilistic mapping from states to actions. The main objective of a RL algorithm is to find an optimal policy. In order to achieve this goal, a key step in many algorithms is to calculate the value function of a given policy  $\pi$ , i.e.,  $V^\pi : \mathcal{S} \rightarrow \mathbb{R}$ , a process known as *policy evaluation*. It is known that  $V^\pi$  is the unique fixed-point of the *Bellman operator*  $T^\pi$ , i.e.,

$$V^\pi = T^\pi V^\pi = R^\pi + \gamma P^\pi V^\pi, \quad (1)$$

where  $R^\pi$  and  $P^\pi$  are the reward function and transition kernel of the Markov chain induced by policy  $\pi$ . In Eq. 1, we may imagine  $V^\pi$  as a  $|\mathcal{S}|$ -dimensional vector and write everything in vector/matrix form. In the following, to simplify the notation, we often drop the dependence of  $T^\pi$ ,  $V^\pi$ ,  $R^\pi$ , and  $P^\pi$  to  $\pi$ .

We denote by  $\pi_b$ , the behavior policy that generates the data, and by  $\pi$ , the target policy that we would like to evaluate. They are the same in the on-policy setting and different in the off-policy scenario. For each state-action pair  $(s_i, a_i)$ , such that  $\pi_b(a_i|s_i) > 0$ , we define the importance-weighting factor  $\rho_i = \pi(a_i|s_i)/\pi_b(a_i|s_i)$  with  $\rho_{\max} \geq 0$  being its maximum value over the state-action pairs.

When  $\mathcal{S}$  is large or infinite, we often use a linear approximation architecture for  $V^\pi$  with parameters  $\theta \in$

$\mathbb{R}^d$  and  $L$ -bounded basis functions  $\{\varphi_i\}_{i=1}^d$ , i.e.,  $\varphi_i : \mathcal{S} \rightarrow \mathbb{R}$  and  $\max_i \|\varphi_i\|_\infty \leq L$ . We denote by  $\phi(\cdot) = (\varphi_1(\cdot), \dots, \varphi_d(\cdot))^\top$  the feature vector and by  $\mathcal{F}$  the linear function space spanned by the basis functions  $\{\varphi_i\}_{i=1}^d$ , i.e.,  $\mathcal{F} = \{f_\theta \mid \theta \in \mathbb{R}^d \text{ and } f_\theta(\cdot) = \phi(\cdot)^\top \theta\}$ . We may write the approximation of  $V$  in  $\mathcal{F}$  in the vector form as  $\hat{v} = \Phi\theta$ , where  $\Phi$  is the  $|\mathcal{S}| \times d$  feature matrix. When only  $n$  training samples of the form  $\mathcal{D} = \{(s_i, a_i, r_i = r(s_i, a_i), s'_i)\}_{i=1}^n$ ,  $s_i \sim \xi$ ,  $a_i \sim \pi_b(\cdot|s_i)$ ,  $s'_i \sim P(\cdot|s_i, a_i)$ , are available ( $\xi$  is a distribution over the state space  $\mathcal{S}$ ), we may write the *empirical Bellman operator*  $\hat{T}$  for a function in  $\mathcal{F}$  as

$$\hat{T}(\hat{\Phi}\theta) = \hat{R} + \gamma \hat{\Phi}'\theta, \quad (2)$$

where  $\hat{\Phi}$  (resp.  $\hat{\Phi}'$ ) is the empirical feature matrix of size  $n \times d$ , whose  $i$ -th row is the feature vector  $\phi(s_i)^\top$  (resp.  $\phi(s'_i)^\top$ ), and  $\hat{R} \in \mathbb{R}^n$  is the reward vector, whose  $i$ -th element is  $r_i$ . We denote by  $\delta_i(\theta) = r_i + \gamma \phi'_i{}^\top \theta - \phi_i^\top \theta$ , the TD error for the  $i$ -th sample  $(s_i, r_i, s'_i)$  and define  $\Delta\phi_i = \phi_i - \gamma \phi'_i$ . Finally, we define the matrices  $A$  and  $C$ , and the vector  $b$  as

$$A := \mathbb{E}[\rho_i \phi_i (\Delta\phi_i)^\top], \quad b := \mathbb{E}[\rho_i \phi_i r_i], \quad C := \mathbb{E}[\phi_i \phi_i^\top], \quad (3)$$

where the expectations are w.r.t.  $\xi$  and  $P^{\pi_b}$ . We also denote by  $\Xi$ , the diagonal matrix whose elements are  $\xi(s)$ , and  $\xi_{\max} := \max_s \xi(s)$ . For each sample  $i$  in the training set  $\mathcal{D}$ , we can calculate an unbiased estimate of  $A$ ,  $b$ , and  $C$  as follows:

$$\hat{A}_i := \rho_i \phi_i \Delta\phi_i^\top, \quad \hat{b}_i := \rho_i r_i \phi_i, \quad \hat{C}_i := \phi_i \phi_i^\top. \quad (4)$$

### 2.1 GRADIENT-BASED TD ALGORITHMS

The class of gradient-based TD (GTD) algorithms were proposed by Sutton *et al.* [2008, 2009]. These algorithms target two objective functions: the *norm of the expected TD update* (NEU) and the *mean-square projected Bellman error* (MSPBE), defined as (see e.g., Maei 2011)<sup>1</sup>

$$\text{NEU}(\theta) = \|\Phi^\top \Xi (T\hat{v} - \hat{v})\|^2, \quad (5)$$

$$\text{MSPBE}(\theta) = \|\hat{v} - \Pi T\hat{v}\|_\xi^2 = \|\Phi^\top \Xi (T\hat{v} - \hat{v})\|_{C^{-1}}^2, \quad (6)$$

where  $C = \mathbb{E}[\phi_i \phi_i^\top] = \Phi^\top \Xi \Phi$  is the covariance matrix defined in Eq. 3 and is assumed to be non-singular, and  $\Pi = \Phi(\Phi^\top \Xi \Phi)^{-1} \Phi^\top \Xi$  is the orthogonal projection operator into the function space  $\mathcal{F}$ , i.e., for any bounded function  $g$ ,  $\Pi g = \arg \min_{f \in \mathcal{F}} \|g - f\|_\xi$ . From (5) and (6), it is clear that NEU and MSPBE are square unweighted and weighted by  $C^{-1}$ ,  $\ell_2$ -norms of the quantity  $\Phi^\top \Xi (T\hat{v} - \hat{v})$ , respectively, and thus, the two objective functions can be unified as

$$J(\theta) = \|\Phi^\top \Xi (T\hat{v} - \hat{v})\|_{M^{-1}}^2 = \|\mathbb{E}[\rho_i \delta_i(\theta) \phi_i]\|_{M^{-1}}^2, \quad (7)$$

<sup>1</sup> It is important to note that  $T$  in (5) and (6) is  $T^\pi$ , the Bellman operator of the target policy  $\pi$ .

with  $M$  equals to the identity matrix  $I$  for NEU and to the covariance matrix  $C$  for MSPBE. The second equality in (7) holds because of the following lemma from Section 4.2 in Maei [2011].

**Lemma 1.** *Let  $\mathcal{D} = \{(s_i, a_i, r_i, s'_i)\}_{i=1}^n$ ,  $s_i \sim \xi$ ,  $a_i \sim \pi_b(\cdot|s_i)$ ,  $s'_i \sim P(\cdot|s_i, a_i)$  be a training set generated by the behavior policy  $\pi_b$  and  $T$  be the Bellman operator of the target policy  $\pi$ . Then, we have*

$$\Phi^\top \Xi(T\hat{v} - \hat{v}) = \mathbb{E}[\rho_i \delta_i(\theta) \phi_i] = b - A\theta.$$

Motivated by minimizing the NEU and MSPBE objective functions using the stochastic gradient methods, the GTD and GTD2 algorithms were proposed with the following update rules:

$$\begin{aligned} \text{GTD:} \quad y_{t+1} &= y_t + \alpha_t (\rho_t \delta_t(\theta_t) \phi_t - y_t), \\ \theta_{t+1} &= \theta_t + \alpha_t \rho_t \Delta \phi_t (y_t^\top \phi_t), \end{aligned} \quad (8)$$

$$\begin{aligned} \text{GTD2:} \quad y_{t+1} &= y_t + \alpha_t (\rho_t \delta_t(\theta_t) - \phi_t^\top y_t) \phi_t, \\ \theta_{t+1} &= \theta_t + \alpha_t \rho_t \Delta \phi_t (y_t^\top \phi_t). \end{aligned} \quad (9)$$

However, it has been shown that the above update rules do not update the value function parameter  $\theta$  in the gradient direction of NEU and MSPBE, and thus, NEU and MSPBE are not the true objective functions of the GTD and GTD2 algorithms [Szepesvári, 2010]. Consider the NEU objective function in (5). Taking its gradient w.r.t.  $\theta$ , we obtain

$$\begin{aligned} -\frac{1}{2} \nabla \text{NEU}(\theta) &= -(\nabla \mathbb{E}[\rho_i \delta_i(\theta) \phi_i^\top]) \mathbb{E}[\rho_i \delta_i(\theta) \phi_i] \\ &= -(\mathbb{E}[\rho_i \nabla \delta_i(\theta) \phi_i^\top]) \mathbb{E}[\rho_i \delta_i(\theta) \phi_i] \\ &= \mathbb{E}[\rho_i \Delta \phi_i \phi_i^\top] \mathbb{E}[\rho_i \delta_i(\theta) \phi_i]. \end{aligned} \quad (10)$$

If the gradient can be written as a single expectation, then it is straightforward to use a stochastic gradient method. However, we have a product of two expectations in (10), and unfortunately, due to the correlation between them, the sample product (with a single sample) won't be an unbiased estimate of the gradient. To tackle this, the GTD algorithm uses an auxiliary variable  $y_t$  to estimate  $\mathbb{E}[\rho_i \delta_i(\theta) \phi_i]$ , and thus, the overall algorithm is no longer a true stochastic gradient method w.r.t. NEU. It can be easily shown that the same problem exists for GTD2 w.r.t. the MSPBE objective function. This prevents us from using the standard convergence analysis techniques of stochastic gradient descent methods to obtain a finite-sample performance bound for the GTD and GTD2 algorithms.

It should be also noted that in the original publications of GTD/GTD2 algorithms [Sutton *et al.*, 2008, 2009], the authors discussed handling the off-policy scenario using both importance and rejected sampling. In rejected sampling that was mainly used in Sutton *et al.* [2008, 2009], a sample  $(s_i, a_i, r_i, s'_i)$  is rejected and the parameter  $\theta$  does not update for this sample, if  $\pi(a_i|s_i) = 0$ . This sampling strategy is not efficient since a lot of samples will be discarded if  $\pi_b$  and  $\pi$  are very different.

## 2.2 RELATED WORK

Before we present a finite-sample performance bound for GTD and GTD2, it would be helpful to give a brief overview of the existing literature on finite-sample analysis of the TD algorithms. The convergence rate of the TD algorithms mainly depends on  $(d, n, \nu)$ , where  $d$  is the size of the approximation space (the dimension of the feature vector),  $n$  is the number of samples, and  $\nu$  is the smallest eigenvalue of the sample-based covariance matrix  $\hat{C} = \hat{\Phi}^\top \hat{\Phi}$ , i.e.,  $\nu = \lambda_{\min}(\hat{C})$ .

Antos *et al.* [2008] proved an error bound of  $O(\frac{d \log d}{n^{1/4}})$  for LSTD in bounded spaces. Lazaric *et al.* [2010b] proposed a LSTD analysis in leaner spaces and obtained a tighter bound of  $O(\sqrt{\frac{d \log d}{n\nu}})$  and later used it to derive a bound for the least-squares policy iteration (LSPI) algorithm [Lazaric *et al.*, 2012]. Tagorti and Scherrer [2014] recently proposed the first convergence analysis for LSTD( $\lambda$ ) and derived a bound of  $\tilde{O}(d/\nu\sqrt{n})$ . The analysis is a bit different than the one in Lazaric *et al.* [2010b] and the bound is weaker in terms of  $d$  and  $\nu$ . Another recent result is by Prashanth *et al.* [2014] that use stochastic approximation to solve LSTD(0), where the resulting algorithm is exactly TD(0) with random sampling (samples are drawn i.i.d. and not from a trajectory), and report a Markov design bound (the bound is computed only at the states used by the algorithm) of  $O(\sqrt{\frac{d}{n\nu}})$  for LSTD(0). All these results are for the on-policy setting, except the one by Antos *et al.* [2008] that also holds for the off-policy formulation. Another work in the off-policy setting is by Ávila Pires and Szepesvári [2012] that uses a bounding trick and improves the result of Antos *et al.* [2008] by a  $\log d$  factor.

The line of research reported here has much in common with work on proximal reinforcement learning [Mahadevan *et al.*, 2014], which explores first-order reinforcement learning algorithms using *mirror maps* [Bubeck, 2014; Juditsky *et al.*, 2008] to construct primal-dual spaces. This work began originally with a dual space formulation of first-order sparse TD learning [Mahadevan and Liu, 2012]. A saddle point formulation for off-policy TD learning was initially explored in Liu *et al.* [2012], where the objective function is the norm of the approximation residual of a linear inverse problem [Ávila Pires and Szepesvári, 2012]. A sparse off-policy GTD2 algorithm with regularized dual averaging is introduced by Qin and Li [2014]. These studies provide different approaches to formulating the problem, first as a variational inequality problem [Juditsky *et al.*, 2008; Mahadevan *et al.*, 2014] or as a linear inverse problem [Liu *et al.*, 2012], or as a quadratic objective function (MSPBE) using two-time-scale solvers [Qin and Li, 2014]. In this paper, we are going to explore the true nature of the GTD algorithms as stochastic gradient algorithm w.r.t the convex-concave saddle-point formulations of NEU and MSPBE.

### 3 SADDLE-POINT FORMULATION OF GTD ALGORITHMS

In this section, we show how the GTD and GTD2 algorithms can be formulated as true stochastic gradient (SG) algorithms by writing their respective objective functions, NEU and MSPBE, in the form of a convex-concave saddle-point. As discussed earlier, this new formulation of GTD and GTD2 as true SG methods allows us to use the convergence analysis techniques for SGs in order to derive finite-sample performance bounds for these RL algorithms. Moreover, it allows us to use more efficient algorithms that have been recently developed to solve SG problems, such as *stochastic Mirror-Prox* (SMP) [Juditsky *et al.*, 2008], to derive more efficient versions of GTD and GTD2.

A particular type of convex-concave saddle-point formulation is formally defined as

$$\min_{\theta} \max_y (L(\theta, y) = \langle b - A\theta, y \rangle + F(\theta) - K(y)), \quad (11)$$

where  $F(\theta)$  is a convex function and  $K(y)$  is a smooth convex function such that

$$K(y) - K(x) - \langle \nabla K(x), y - x \rangle \leq \frac{L_K}{2} \|x - y\|^2. \quad (12)$$

Next we follow Juditsky *et al.* [2008]; Nemirovski *et al.* [2009]; Chen *et al.* [2013] and define the following error function for the saddle-point problem (11).

**Definition 1.** *The error function of the saddle-point problem (11) at each point  $(\theta', y')$  is defined as*

$$\text{Err}(\theta', y') = \max_y L(\theta', y) - \min_{\theta} L(\theta, y'). \quad (13)$$

In this paper, we consider the saddle-point problem (11) with  $F(\theta) = 0$  and  $K(y) = \frac{1}{2} \|y\|_M^2$ , i.e.,

$$\min_{\theta} \max_y \left( L(\theta, y) = \langle b - A\theta, y \rangle - \frac{1}{2} \|y\|_M^2 \right), \quad (14)$$

where  $A$  and  $b$  were defined by Eq. 3, and  $M$  is a positive definite matrix. It is easy to show that  $K(y) = \frac{1}{2} \|y\|_M^2$  satisfies the condition in Eq. 12.

We first show in Proposition 1 that if  $(\theta^*, y^*)$  is the saddle-point of problem (14), then  $\theta^*$  will be the optimum of NEU and MSPBE defined in Eq. 7. We then prove in Proposition 2 that GTD and GTD2 in fact find this saddle-point.

**Proposition 1.** *For any fixed  $\theta$ , we have  $\frac{1}{2} J(\theta) = \max_y L(\theta, y)$ , where  $J(\theta)$  is defined by Eq. 7.*

*Proof.* Since  $L(\theta, y)$  is an unconstrained quadratic program w.r.t.  $y$ , the optimal  $y^*(\theta) = \arg \max_y L(\theta, y)$  can be analytically computed as

$$y^*(\theta) = M^{-1}(b - A\theta). \quad (15)$$

The result follows by plugging  $y^*$  into (14) and using the definition of  $J(\theta)$  in Eq. 7 and Lemma 1.  $\square$

**Proposition 2.** *GTD and GTD2 are true stochastic gradient algorithms w.r.t. the objective function  $L(\theta, y)$  of the saddle-point problem (14) with  $M = I$  and  $M = C = \Phi^\top \Xi \Phi$  (the covariance matrix), respectively.*

*Proof.* It is easy to see that the gradient updates of the saddle-point problem (14) (ascending in  $y$  and descending in  $\theta$ ) may be written as

$$\begin{aligned} y_{t+1} &= y_t + \alpha_t (b - A\theta_t - My_t), \\ \theta_{t+1} &= \theta_t + \alpha_t A^\top y_t. \end{aligned} \quad (16)$$

We denote  $\hat{M} := I$  (resp.  $\hat{M} := \hat{C}$ ) for GTD (resp. GTD2). We may obtain the update rules of GTD and GTD2 by replacing  $A$ ,  $b$ , and  $C$  in (16) with their unbiased estimates  $\hat{A}$ ,  $\hat{b}$ , and  $\hat{C}$  from Eq. 4, which completes the proof.  $\square$

### 4 FINITE-SAMPLE ANALYSIS

In this section, we provide a finite-sample analysis for a revised version of the GTD/GTD2 algorithms. We first describe the revised GTD algorithms in Section 4.1 and then dedicate the rest of Section 4 to their sample analysis. Note that from now on we use the  $M$  matrix (and its unbiased estimate  $\hat{M}_t$ ) to have a unified analysis for GTD and GTD2 algorithms. As described earlier,  $M$  is replaced by the identity matrix  $I$  in GTD and by the covariance matrix  $C$  (and its unbiased estimate  $\hat{C}_t$ ) in GTD2.

#### 4.1 THE REVISED GTD ALGORITHMS

The revised GTD algorithms that we analyze in this paper (see Algorithm 1) have three differences with the standard GTD algorithms of Eqs. 8 and 9 (and Eq. 16). **1)** We guarantee that the parameters  $\theta$  and  $y$  remain bounded by projecting them onto bounded convex feasible sets  $\Theta$  and  $Y$  defined in Assumption 2. In Algorithm 1, we denote by  $\Pi_\Theta$  and  $\Pi_Y$ , the projection into sets  $\Theta$  and  $Y$ , respectively. This is standard in stochastic approximation algorithms and has been used in off-policy TD( $\lambda$ ) [Yu, 2012] and actor-critic algorithms (e.g., Bhatnagar *et al.* 2009). **2)** after  $n$  iterations ( $n$  is the number of training samples in  $\mathcal{D}$ ), the algorithms return the weighted (by the step size) average of the parameters at all the  $n$  iterations (see Eq. 18). **3)** The step-size  $\alpha_t$  is selected as described in the proof of Proposition 3 in the supplementary material. Note that this fixed step size of  $O(1/\sqrt{n})$  is required for the high-probability bound in Proposition 3 (see Nemirovski *et al.* 2009 for more details).

#### 4.2 ASSUMPTIONS

In this section, we make several assumptions on the MDP and basis functions that are used in our finite-sample analysis of the revised GTD algorithms. These assumptions are

---

**Algorithm 1** Revised GTD Algorithms
 

---

 1: **for**  $t = 1, \dots, n$  **do**

2: Update parameters

$$\begin{aligned} y_{t+1} &= \Pi_Y \left( y_t + \alpha_t (\hat{b}_t - \hat{A}_t \theta_t - \hat{M}_t y_t) \right) \\ \theta_{t+1} &= \Pi_\Theta \left( \theta_t + \alpha_t \hat{A}_t^\top y_t \right) \end{aligned} \quad (17)$$

 3: **end for**

4: OUTPUT

$$\bar{\theta}_n := \frac{\sum_{t=1}^n \alpha_t \theta_t}{\sum_{t=1}^n \alpha_t}, \quad \bar{y}_n := \frac{\sum_{t=1}^n \alpha_t y_t}{\sum_{t=1}^n \alpha_t} \quad (18)$$


---

quite standard and are similar to those made in the prior work on GTD algorithms [Sutton *et al.*, 2008, 2009; Maei, 2011] and those made in the analysis of SG algorithms [Nemirovski *et al.*, 2009].

**Assumption 2. (Feasibility Sets)** We define the bounded closed convex sets  $\Theta \subset \mathbb{R}^d$  and  $Y \subset \mathbb{R}^d$  as the feasible sets in Algorithm 1. We further assume that the saddle-point  $(\theta^*, y^*)$  of the optimization problem (14) belongs to  $\Theta \times Y$ .

We also define  $D_\theta := [\max_{\theta \in \Theta} \|\theta\|_2^2 - \min_{\theta \in \Theta} \|\theta\|_2^2]^{1/2}$ ,  $D_y := [\max_{y \in Y} \|y\|_2^2 - \min_{y \in Y} \|y\|_2^2]^{1/2}$ , and  $R = \max \{ \max_{\theta \in \Theta} \|\theta\|_2, \max_{y \in Y} \|y\|_2 \}$ .

**Assumption 3. (Non-singularity)** We assume that the covariance matrix  $C = \mathbb{E}[\phi_i \phi_i^\top]$  and matrix  $A = \mathbb{E}[\rho_i \phi_i (\Delta \phi_i)^\top]$  are non-singular.

**Assumption 4. (Boundedness)** Assume the features  $(\phi_i, \phi'_i)$  have uniformly bounded second moments. This together with the boundedness of features (by  $L$ ) and importance weights (by  $\rho_{\max}$ ) guarantees that the matrices  $A$  and  $C$ , and vector  $b$  are uniformly bounded.

This assumption guarantees that for any  $(\theta, y) \in \Theta \times Y$ , the unbiased estimators of  $b - A\theta - My$  and  $A^\top y$ , i.e.,

$$\begin{aligned} \mathbb{E}[\hat{b}_t - \hat{A}_t \theta - \hat{M}_t y] &= b - A\theta - My, \\ \mathbb{E}[\hat{A}_t^\top y] &= A^\top y, \end{aligned} \quad (19)$$

all have bounded variance, i.e.,

$$\begin{aligned} \mathbb{E}[|\hat{b}_t - \hat{A}_t \theta - \hat{M}_t y - (b - A\theta - My)|^2] &\leq \sigma_1^2, \\ \mathbb{E}[|\hat{A}_t^\top y - A^\top y|^2] &\leq \sigma_2^2, \end{aligned} \quad (20)$$

where  $\sigma_1$  and  $\sigma_2$  are non-negative constants. We further define

$$\sigma^2 = \sigma_1^2 + \sigma_2^2. \quad (21)$$

Assumption 4 also gives us the following ‘‘light-tail’’ assumption. There exist constants  $M_{*,\theta}$  and  $M_{*,y}$  such that

$$\begin{aligned} \mathbb{E}[\exp\{\frac{|\hat{b}_t - \hat{A}_t \theta - \hat{M}_t y|^2}{M_{*,\theta}^2}\}] &\leq \exp\{1\}, \\ \mathbb{E}[\exp\{\frac{|\hat{A}_t^\top y|^2}{M_{*,y}^2}\}] &\leq \exp\{1\}. \end{aligned} \quad (22)$$

This ‘‘light-tail’’ assumption is equivalent to the assumption in Eq. 3.16 in Nemirovski *et al.* [2009] and is necessary for the high-probability bound of Proposition 3. We will show how to compute  $M_{*,\theta}$ ,  $M_{*,y}$  in the Appendix.

### 4.3 FINITE-SAMPLE PERFORMANCE BOUNDS

The finite-sample performance bounds that we derive for the GTD algorithms in this section are for the case that the training set  $\mathcal{D}$  has been generated as discussed in Section 2. We further discriminate between the on-policy ( $\pi = \pi_b$ ) and off-policy ( $\pi \neq \pi_b$ ) scenarios. The sampling scheme used to generate  $\mathcal{D}$ , in which the first state of each tuple,  $s_i$ , is an i.i.d. sample from a distribution  $\xi$ , also considered in the original GTD and GTD2 papers, for the analysis of these algorithms, and not in the experiments [Sutton *et al.*, 2008, 2009]. Another scenario that can motivate this sampling scheme is when we are given a set of high-dimensional data generated either in an on-policy or off-policy manner, and  $d$  is so large that the value function of the target policy cannot be computed using a least-squares method (that involves matrix inversion), and iterative techniques similar to GTD/GTD2 are required.

We first derive a high-probability bound on the error function of the saddle-point problem (14) at the GTD solution  $(\bar{\theta}_n, \bar{y}_n)$ . Before stating this result in Proposition 3, we report the following lemma that is used in its proof.

**Lemma 2.** *The induced  $\ell_2$ -norm of matrix  $A$  and the  $\ell_2$ -norm of vector  $b$  are bounded by*

$$\|A\|_2 \leq (1 + \gamma)\rho_{\max}L^2d, \quad \|b\|_2 \leq \rho_{\max}LR_{\max}. \quad (23)$$

*Proof.* See the supplementary material.  $\square$

**Proposition 3.** *Let  $(\bar{\theta}_n, \bar{y}_n)$  be the output of the GTD algorithm after  $n$  iterations (see Eq. 18). Then, with probability at least  $1 - \delta$ , we have*

$$\begin{aligned} \text{Err}(\bar{\theta}_n, \bar{y}_n) &\leq \sqrt{\frac{5}{n}}(8 + 2 \log \frac{2}{\delta})R^2 \\ &\times \left( \rho_{\max}L \left( 2(1 + \gamma)Ld + \frac{R_{\max}}{R} \right) + \tau + \frac{\sigma}{R} \right), \end{aligned} \quad (24)$$

where  $\text{Err}(\bar{\theta}_n, \bar{y}_n)$  is the error function of the saddle-point problem (14) defined by Eq. 13,  $R$  defined in Assumption 2,  $\sigma$  is from Eq. 21, and  $\tau = \sigma_{\max}(M)$  is the largest singular value of  $M$ , which means  $\tau = 1$  for GTD and  $\tau = \sigma_{\max}(C)$  for GTD2.

*Proof.* See the supplementary material.  $\square$

**Theorem 1.** *Let  $\bar{\theta}_n$  be the output of the GTD algorithm after  $n$  iterations (see Eq. 18). Then, with probability at least  $1 - \delta$ , we have*

$$\frac{1}{2} \|A\bar{\theta}_n - b\|_\xi^2 \leq \tau \xi_{\max} \text{Err}(\bar{\theta}_n, \bar{y}_n). \quad (25)$$

*Proof.* From Proposition 1, for any  $\theta$ , we have

$$\max_y L(\theta, y) = \frac{1}{2} \|A\theta - b\|_{M^{-1}}^2.$$

Given Assumption 3, the system of linear equations  $A\theta = b$  has a solution  $\theta^*$ , i.e., the (off-policy) fixed-point  $\theta^*$  exists, and thus, we may write

$$\begin{aligned} \min_{\theta} \max_y L(\theta, y) &= \min_{\theta} \frac{1}{2} \|A\theta - b\|_{M^{-1}}^2 \\ &= \frac{1}{2} \|A\theta^* - b\|_{M^{-1}}^2 = 0. \end{aligned}$$

In this case, we also have<sup>2</sup>

$$\begin{aligned} \min_{\theta} L(\theta, y) &\leq \max_y \min_{\theta} L(\theta, y) \leq \min_{\theta} \max_y L(\theta, y) \\ &= \frac{1}{2} \|A\theta^* - b\|_{M^{-1}}^2 = 0. \end{aligned} \quad (26)$$

From Eq. 26, for any  $(\theta, y) \in \Theta \times Y$  including  $(\bar{\theta}_n, \bar{y}_n)$ , we may write

$$\begin{aligned} \text{Err}(\bar{\theta}_n, \bar{y}_n) &= \max_y L(\bar{\theta}_n, y) - \min_{\theta} L(\theta, \bar{y}_n) \\ &\geq \max_y L(\bar{\theta}_n, y) = \frac{1}{2} \|A\bar{\theta}_n - b\|_{M^{-1}}^2. \end{aligned} \quad (27)$$

Since  $\|A\bar{\theta}_n - b\|_{\xi}^2 \leq \tau \xi_{\max} \|A\bar{\theta}_n - b\|_{M^{-1}}^2$ , where  $\tau$  is the largest singular value of  $M$ , we have

$$\frac{1}{2} \|A\bar{\theta}_n - b\|_{\xi}^2 \leq \frac{\tau \xi_{\max}}{2} \|A\bar{\theta}_n - b\|_{M^{-1}}^2 \leq \tau \xi_{\max} \text{Err}(\bar{\theta}_n, \bar{y}_n). \quad (28)$$

The proof follows by combining Eqs. 28 and Proposition 3. It completes the proof.  $\square$

With the results of Proposition 3 and Theorem 1, we are now ready to derive finite-sample bounds on the performance of GTD/GTD2 in both on-policy and off-policy settings.

### 4.3.1 On-Policy Performance Bound

In this section, we consider the on-policy setting in which the behavior and target policies are equal, i.e.,  $\pi_b = \pi$ , and the sampling distribution  $\xi$  is the stationary distribution of the target policy  $\pi$  (and the behavior policy  $\pi_b$ ). We use Lemma 3 to derive our on-policy bound. The proof of this lemma can be found in Geist *et al.* [2012].

**Lemma 3.** *For any parameter vector  $\theta$  and corresponding  $\hat{v} = \Phi\theta$ , the following equality holds*

$$V - \hat{v} = (I - \gamma \Pi P)^{-1} [(V - \Pi V) + \Phi C^{-1}(b - A\theta)]. \quad (29)$$

Using Lemma 3, we derive the following performance bound for GTD/GTD2 in the on-policy setting.

<sup>2</sup>We may write the second inequality as an equality for our saddle-point problem defined by Eq. 14.

**Proposition 4.** *Let  $V$  be the value of the target policy and  $\bar{v}_n = \Phi\bar{\theta}_n$ , where  $\bar{\theta}_n$  defined by (18), be the value function returned by on-policy GTD/GTD2. Then, with probability at least  $1 - \delta$ , we have*

$$\|V - \bar{v}_n\|_{\xi} \leq \frac{1}{1 - \gamma} \left( \|V - \Pi V\|_{\xi} + \frac{L}{\nu} \sqrt{2d\tau \xi_{\max} \text{Err}(\bar{\theta}_n, \bar{y}_n)} \right) \quad (30)$$

where  $\text{Err}(\bar{\theta}_n, \bar{y}_n)$  is upper-bounded by Eq. 24 in Proposition 3, with  $\rho_{\max} = 1$  (on-policy setting).

*Proof.* See the supplementary material.  $\square$

**Remark:** It is important to note that Proposition 4 shows that the error in the performance of the GTD/GTD2 algorithm in the on-policy setting is of  $O\left(\frac{L^2 d \sqrt{\tau \xi_{\max} \log \frac{1}{\delta}}}{n^{1/4} \nu}\right)$ . Also note that the term  $\frac{\tau}{\nu}$  in the GTD2 bound is the conditioning number of the covariance matrix  $C$ .

### 4.3.2 Off-Policy Performance Bound

In this section, we consider the off-policy setting in which the behavior and target policies are different, i.e.,  $\pi_b \neq \pi$ , and the sampling distribution  $\xi$  is the stationary distribution of the behavior policy  $\pi_b$ . We assume that off-policy fixed-point solution exists, i.e., there exists a  $\theta^*$  satisfying  $A\theta^* = b$ . Note that this is a direct consequence of Assumption 3 in which we assumed that the matrix  $A$  in the off-policy setting is non-singular. We use Lemma 4 to derive our off-policy bound. The proof of this lemma can be found in Kolter [2011]. Note that  $\kappa(\bar{D})$  in his proof is equal to  $\sqrt{\rho_{\max}}$  in our paper.

**Lemma 4.** *If  $\Xi$  satisfies the following linear matrix inequality*

$$\begin{bmatrix} \Phi^{\top} \Xi \Phi & \Phi^{\top} \Xi P \Phi \\ \Phi^{\top} P^{\top} \Xi \Phi & \Phi^{\top} \Xi \Phi \end{bmatrix} \succeq 0 \quad (31)$$

and let  $\theta^*$  be the solution to  $A\theta^* = b$ , then we have

$$\|V - \Phi\theta^*\|_{\xi} \leq \frac{1 + \gamma \sqrt{\rho_{\max}}}{1 - \gamma} \|V - \Pi V\|_{\xi}. \quad (32)$$

Note that the condition on  $\Xi$  in Eq. 31 guarantees that the behavior and target policies are not too far away from each other. Using Lemma 4, we derive the following performance bound for GTD/GTD2 in the off-policy setting.

**Proposition 5.** *Let  $V$  be the value of the target policy and  $\bar{v}_n = \Phi\bar{\theta}_n$ , where  $\bar{\theta}_n$  is defined by (18), be the value function returned by off-policy GTD/GTD2. Also let the sampling distribution  $\Xi$  satisfies the condition in Eq. 31. Then,*

with probability at least  $1 - \delta$ , we have

$$\|V - \bar{v}_n\|_\xi \leq \frac{1 + \gamma\sqrt{\rho_{\max}}}{1 - \gamma} \|V - \Pi V\|_\xi \quad (33)$$

$$+ \sqrt{\frac{2\tau_C\tau\xi_{\max}}{\sigma_{\min}(A^\top M^{-1}A)} \text{Err}(\bar{\theta}_n, \bar{y}_n)},$$

where  $\tau_C = \sigma_{\max}(C)$ .

*Proof.* See the supplementary material.  $\square$

## 5 ACCELERATED ALGORITHM

As discussed at the beginning of Section 3, this saddle-point formulation not only gives us the opportunity to use the techniques for the analysis of SG methods to derive finite-sample performance bounds for the GTD algorithms, as we will show in Section 4, but also it allows us to use the powerful algorithms that have been recently developed to solve the SG problems and derive more efficient versions of GTD and GTD2. Stochastic Mirror-Prox (SMP) [Juditsky *et al.*, 2008] is an ‘‘almost dimension-free’’ non-Euclidean extra-gradient method that deals with both smooth and non-smooth stochastic optimization problems (see Juditsky and Nemirovski 2011 and Bubeck 2014 for more details). Using SMP, we propose a new version of GTD/GTD2, called GTD-MP/GTD2-MP, with the following update formula.<sup>3</sup>

$$y_t^m = y_t + \alpha_t(\hat{b}_t - \hat{A}_t\theta_t - \hat{M}_ty_t), \quad \theta_t^m = \theta_t + \alpha_t\hat{A}_t^\top y_t,$$

$$y_{t+1} = y_t + \alpha_t(\hat{b}_t - \hat{A}_t\theta_t^m - \hat{M}_ty_t^m), \quad \theta_{t+1} = \theta_t + \alpha_t\hat{A}_t^\top y_t^m.$$

After  $T$  iterations, these algorithms return  $\bar{\theta}_T := \frac{\sum_{t=1}^T \alpha_t \theta_t}{\sum_{t=1}^T \alpha_t}$  and  $\bar{y}_T := \frac{\sum_{t=1}^T \alpha_t y_t}{\sum_{t=1}^T \alpha_t}$ . The details of the algorithm is shown in Algorithm 2, and the experimental comparison study between GTD2 and GTD2-MP is reported in Section 7.

## 6 FURTHER ANALYSIS

### 6.1 ACCELERATION ANALYSIS

In this section, we are going to discuss the convergence rate of the accelerated algorithms using off-the-shelf accelerated solvers for saddle-point problems. For simplicity, we will discuss the error bound of  $\frac{1}{2}\|A\theta - b\|_{M^{-1}}^2$ , and the corresponding error bound of  $\frac{1}{2}\|A\theta - b\|_\xi^2$  and  $\|V - \bar{v}_n\|_\xi$  can be likewise derived as in above analysis. As can be seen from the above analysis, the convergence rate of the GTD algorithms family is

$$\text{(GTD/GTD2)} : \quad O\left(\frac{\tau + \|A\|_2 + \sigma}{\sqrt{n}}\right) \quad (35)$$

<sup>3</sup>For simplicity, we only describe mirror-prox GTD methods where the mirror map is identity, which can also be viewed as extragradient (EG) GTD methods. Mahadevan *et al.* [2014] gives a more detailed discussion of a broad range of mirror maps in RL.

---

### Algorithm 2 GTD2-MP

---

1: **for**  $t = 1, \dots, n$  **do**

2: Update parameters

$$\delta_t = r_t - \theta_t^\top \Delta \phi_t$$

$$y_t^m = y_t + \alpha_t(\rho_t \delta_t - \phi_t^\top y_t) \phi_t$$

$$\theta_t^m = \theta_t + \alpha_t \rho_t \Delta \phi_t (\phi_t^\top y_t)$$

$$\delta_t^m = r_t - (\theta_t^m)^\top \Delta \phi_t$$

$$y_{t+1} = y_t + \alpha_t(\rho_t \delta_t^m - \phi_t^\top y_t^m) \phi_t$$

$$\theta_{t+1} = \theta_t + \alpha_t \rho_t \Delta \phi_t (\phi_t^\top y_t^m)$$

3: **end for**

4: OUTPUT

$$\bar{\theta}_n := \frac{\sum_{t=1}^n \alpha_t \theta_t}{\sum_{t=1}^n \alpha_t}, \quad \bar{y}_n := \frac{\sum_{t=1}^n \alpha_t y_t}{\sum_{t=1}^n \alpha_t} \quad (34)$$


---

In this section, we raise an interesting question: what is the ‘‘optimal’’ GTD algorithm? To answer this question, we review the convex-concave formulation of GTD2. According to convex programming complexity theory [Juditsky *et al.*, 2008], the un-improvable convergence rate of stochastic saddle-point problem (14) is

$$\text{(Optimal)} : \quad O\left(\frac{\tau}{n^2} + \frac{\|A\|_2}{n} + \frac{\sigma}{\sqrt{n}}\right) \quad (36)$$

There are many readily available stochastic saddle-point solvers, such as stochastic Mirror-Prox (SMP) [Juditsky *et al.*, 2008] algorithm, which leads to our proposed GTD2-MP algorithm. SMP is able to accelerate the convergence rate of our gradient TD method to:

$$\text{(SMP)} : \quad O\left(\frac{\tau + \|A\|_2}{n} + \frac{\sigma}{\sqrt{n}}\right), \quad (37)$$

and stochastic accelerated primal-dual (SAPD) method [Chen *et al.*, 2013] which can reach the optimal convergence rate in (36). Due to space limitations, we are unable to present a more complete description, and refer interested readers to Juditsky *et al.* [2008]; Chen *et al.* [2013] for more details.

### 6.2 LEARNING WITH BIASED $\rho_t$

The importance weight factor  $\rho_t$  is lower bounded by 0, but yet may have an arbitrarily large upper bound. In real applications, the importance weight factor  $\rho_t$  may not be estimated exactly, i.e., the estimation  $\hat{\rho}_t$  is a biased estimation of the true  $\rho_t$ . To this end, the stochastic gradient we obtained is not unbiased gradient of  $L(\theta, y)$  anymore. This falls into a broad category of learning with inexact stochastic gradient, or termed as stochastic gradient methods with an inexact oracle [Devolder, 2011]. Given the inexact stochastic gradient, the convergence rate and performance bound become much worse than the results with



exact stochastic gradient. Based on the analysis by Juditsky *et al.* [2008], we have the error bound for inexact estimation of  $\rho_t$ .

**Proposition 6.** *Let  $\bar{\theta}_n$  be defined as above. Assume at the  $t$ -th iteration,  $\hat{\rho}_t$  is the estimation of the importance weight factor  $\rho_t$  with bounded bias such that  $\mathbb{E}[\hat{\rho}_t - \rho_t] \leq \epsilon$ . The convergence rates of GTD/GTD2 algorithms with iterative averaging is as follows, i.e.,*

$$\|A\bar{\theta}_n - b\|_{M^{-1}}^2 \leq O\left(\frac{\tau + \|A\|_2 + \sigma}{\sqrt{n}}\right) + O(\epsilon) \quad (38)$$

This implies that the inexact estimation of  $\rho_t$  may cause disastrous estimation error, which implies that an exact estimation of  $\rho_t$  is very important.

### 6.3 FINITE-SAMPLE ANALYSIS OF ONLINE LEARNING

Another more challenging scenario is online learning scenario, where the samples are interactively generated by the environment, or by an interactive agent. The difficulty lies in that the sample distribution does not follow i.i.d sampling condition anymore, but follows an underlying Markov chain  $\mathcal{M}$ . If the Markov chain  $\mathcal{M}$ 's mixing time is small enough, i.e., the sample distribution reduces to the stationary distribution of  $\pi_b$  very fast, our analysis still applies. However, it is usually the case that the underlying Markov chain's mixing time  $\tau_{\text{mix}}$  is not small enough. The analysis result can be obtained by extending the result of recent work [Duchi *et al.*, 2012] from strongly convex loss functions to saddle-point problems, which is non-trivial and is thus left for future work.

### 6.4 DISCUSSION OF TDC ALGORITHM

Now we discuss the limitation of our analysis with regard to the temporal difference with correction (TDC) algorithm [Sutton *et al.*, 2009]. Interestingly, the TDC algorithm seems not to have an explicit saddle-point representation, since it incorporates the information of the optimal  $y_t^*(\theta_t)$  into the update of  $\theta_t$ , a quasi-stationary condition which is commonly used in two-time-scale stochastic approximation approaches. An intuitive answer to the advantage of TDC over GTD2 is that the TDC update of  $\theta_t$  can be considered as incorporating the prior knowledge into the update rule: for a stationary  $\theta_t$ , if the optimal  $y_t^*(\theta_t)$  has a closed-form solution or is easy to compute, then incorporating this  $y_t^*(\theta_t)$  into the update law tends to accelerate the algorithm's convergence performance. For the GTD2 update, note that there is a sum of two terms where  $y_t$  appears, which are  $\rho_t(\phi_t - \gamma\phi_t)(y_t^T \phi_t) = \rho_t\phi_t(y_t^T \phi_t) - \gamma\rho_t\phi_t'(y_t^T \phi_t)$ . Replacing  $y_t$  in the first term with  $y_t^*(\theta_t) = \mathbb{E}[\phi_t\phi_t^T]^{-1}\mathbb{E}[\rho_t\delta_t(\theta_t)\phi_t]$ , we have the TDC update rule. Note that in contrast to GTD/GTD2, TDC is a two-time scale algorithm; Also, note that TDC does not minimize

any objective functions and the convergence of TDC requires more restrictions than GTD2 as shown by Sutton *et al.* [2009].

## 7 EMPIRICAL EVALUATION

In this section, we compare the previous GTD2 method with our proposed GTD2-MP method using various domains with regard to their value function approximation performance capability. It should be mentioned that since the major focus of this paper is on policy evaluation, the comparative study focuses on value function approximation and thus comparisons on control learning performance is not reported in this paper.

### 7.1 BAIRD DOMAIN

The Baird example [Baird, 1995] is a well-known example to test the performance of off-policy convergent algorithms. Constant stepsize  $\alpha = 0.005$  for GTD2 and  $\alpha = 0.004$  for GTD2-MP, which are chosen via comparison studies as in [Dann *et al.*, 2014]. Figure 1 shows the MSPBE curve of GTD2, GTD2-MP of 8000 steps averaged over 200 runs. We can see that GTD2-MP has a significant improvement over the GTD2 algorithm wherein both the MSPBE and the variance are substantially reduced.

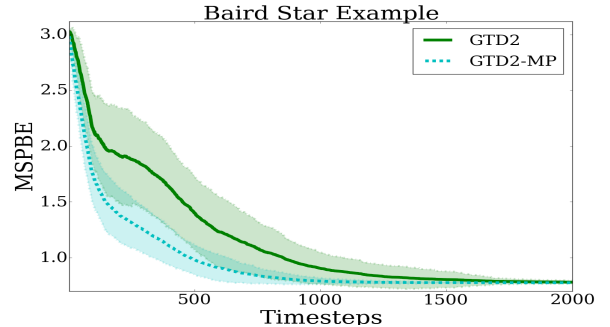


Figure 1: Off-Policy Convergence Comparison

### 7.2 50-STATE CHAIN DOMAIN

The 50 state chain [Lagoudakis and Parr, 2003] is a standard MDP domain. There are 50 discrete states  $\{s_i\}_{i=1}^{50}$  and two actions moving the agent left  $s_i \rightarrow s_{\max(i-1,1)}$  and right  $s_i \rightarrow s_{\min(i+1,50)}$ . The actions succeed with probability 0.9; failed actions move the agent in the opposite direction. The discount factor is  $\gamma = 0.9$ . The agent receives a reward of +1 when in states  $s_{10}$  and  $s_{41}$ . All other states have a reward of 0. In this experiment, we compare the performance of the value approximation w.r.t different set of stepsizes  $\alpha = 0.0001, 0.001, 0.01, 0.1, 0.2, \dots, 0.9$  using the BEBF basis [Parr *et al.*, 2007], and Figure 2 shows the value function approximation result, where the cyan curve

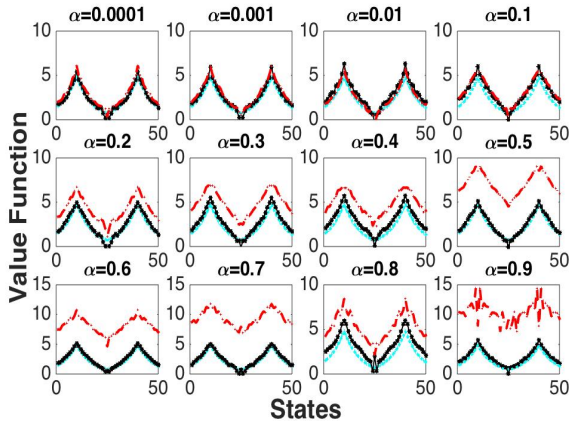


Figure 2: Chain Domain

is the true value function, the red dashed curve is the GTD result, and the black curve is the GTD2-MP result. From the figure, one can see that GTD2-MP is much more robust with stepsize choice than the GTD2 algorithm.

### 7.3 ENERGY MANAGEMENT DOMAIN

In this experiment we compare the performance of the algorithms on an energy management domain. The decision maker must decide how much energy to purchase or sell subject to stochastic prices. This problem is relevant in the context of utilities as well as in settings such as hybrid vehicles. The prices are generated from a Markov chain process. The amount of available storage is limited and it also degrades with use. The degradation process is based on the physical properties of lithium-ion batteries and discourages fully charging or discharging the battery. The energy arbitrage problem is closely related to the broad class of inventory management problems, with the storage level corresponding to the inventory. However, there are no known results describing the structure of optimal threshold policies in energy storage.

Note that since for this off-policy evaluation problem, the formulated  $A\theta = b$  does not have a solution, and thus the optimal  $MSPBE(\theta^*)$  (resp.  $MSBE(\theta^*)$ ) do not reduce to 0. The result is averaged over 200 runs, and  $\alpha = 0.001$  for both GTD2 and GTD2-MP is chosen via comparison studies for each algorithm. As can be seen from Figure 3, in the initial transit state, GTD2-MP performs much better than GTD2 at the transient state. Then after reaching the steady state, as can be seen from Table 1, we can see that GTD2-MP reaches better steady state solution than the GTD algorithm. Based on the above empirical results and many other experiments we have conducted in other domains, we can conclude that GTD2-MP usually performs much better than the “vanilla” GTD2 algorithm.

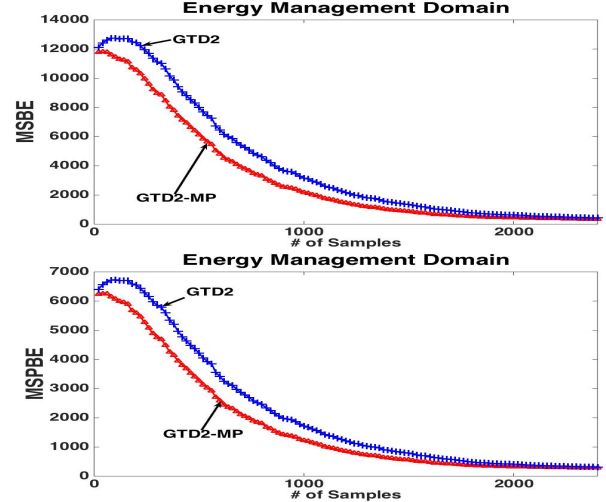


Figure 3: Energy Management Example

| Algorithm | MSPBE | MSBE  |
|-----------|-------|-------|
| GTD2      | 176.4 | 228.7 |
| GTD2-MP   | 138.6 | 191.4 |

Table 1: Steady State Performance Comparison

## 8 SUMMARY

In this paper, we showed how gradient TD methods can be shown to be true stochastic gradient methods with respect to a saddle-point primal-dual objective function, which paved the way for the finite-sample analysis of off-policy convergent gradient-based temporal difference learning algorithms such as GTD and GTD2. Both error bound and performance bound are provided, which shows that the value function approximation bound of the GTD algorithms family is  $O\left(\frac{d}{n^{1/4}}\right)$ . Further, two revised algorithms, namely the projected GTD2 algorithm and the accelerated GTD2-MP algorithm, are proposed. There are many interesting directions for future research. Our framework can be easily used to design regularized sparse gradient off-policy TD methods. One interesting direction is to investigate the convergence rate and performance bound for the TDC algorithm, which lacks a saddle-point formulation. The other is to explore tighter value function approximation bounds for off-policy learning.

## ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant Nos. IIS-1216467. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

## References

- A. Antos, Cs. Szepesvári, and R. Munos. Learning near-optimal policies with Bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning Journal*, 71:89–129, 2008.
- B. Ávila Pires and C. Szepesvári. Statistical linear estimation with penalized estimators: an application to reinforcement learning. In *Proceedings of the 29th International Conference on Machine Learning*, pages 1535–1542, 2012.
- L. C. Baird. Residual algorithms: Reinforcement learning with function approximation. In *International Conference on Machine Learning*, pages 30–37, 1995.
- H. H Bauschke and P. L Combettes. *Convex analysis and monotone operator theory in Hilbert spaces*. Springer, 2011.
- D. Bertsekas and J. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, Belmont, Massachusetts, 1996.
- S. Bhatnagar, R. Sutton, M. Ghavamzadeh, and M. Lee. Natural actor-critic algorithms. *Automatica*, 45(11):2471–2482, 2009.
- V. Borkar. *Stochastic Approximation: A Dynamical Systems Viewpoint*. Cambridge University Press, 2008.
- S. Bubeck. Theory of convex optimization for machine learning. *arXiv:1405.4980*, 2014.
- Y. Chen, G. Lan, and Y. Ouyang. Optimal primal-dual methods for a class of saddle point problems. *arXiv:1309.5548*, 2013.
- C. Dann, G. Neumann, and J. Peters. Policy evaluation with temporal differences: A survey and comparison. *Journal of Machine Learning Research*, 15:809–883, 2014.
- O. Devolder. Stochastic first order methods in smooth convex optimization. Technical report, Université catholique de Louvain, Center for Operations Research and Econometrics, 2011.
- J. Duchi, A. Agarwal, M. Johansson, and M. Jordan. Ergodic mirror descent. *SIAM Journal on Optimization*, 22(4):1549–1578, 2012.
- M. Geist, B. Scherrer, A. Lazaric, and M. Ghavamzadeh. A Dantzig Selector approach to temporal difference learning. In *International Conference on Machine Learning*, pages 1399–1406, 2012.
- M. Ghavamzadeh, A. Lazaric, O. Maillard, and R. Munos. LSTD with Random Projections. In *Proceedings of the International Conference on Neural Information Processing Systems*, pages 721–729, 2010.
- M. Ghavamzadeh, A. Lazaric, R. Munos, and M. Hoffman. Finite-sample analysis of Lasso-TD. In *Proceedings of the 28th International Conference on Machine Learning*, pages 1177–1184, 2011.
- A. Juditsky and A. Nemirovski. *Optimization for Machine Learning*. MIT Press, 2011.
- A. Juditsky, A. Nemirovskii, and C. Tauvel. Solving variational inequalities with stochastic mirror-prox algorithm. *arXiv:0809.0815*, 2008.
- S. Kakade and J. Langford. Approximately optimal approximate reinforcement learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pages 267–274, 2002.
- Z. Kolter. The fixed points of off-policy TD. In *Advances in Neural Information Processing Systems 24*, pages 2169–2177, 2011.
- M. Lagoudakis and R. Parr. Least-squares policy iteration. *Journal of Machine Learning Research*, 4:1107–1149, 2003.
- A. Lazaric, M. Ghavamzadeh, and R. Munos. Analysis of a classification-based policy iteration algorithm. In *Proceedings of the Twenty-Seventh International Conference on Machine Learning*, pages 607–614, 2010.
- A. Lazaric, M. Ghavamzadeh, and R. Munos. Finite-sample analysis of LSTD. In *Proceedings of 27th International Conference on Machine Learning*, pages 615–622, 2010.
- A. Lazaric, M. Ghavamzadeh, and R. Munos. Finite-sample analysis of least-squares policy iteration. *Journal of Machine Learning Research*, 13:3041–3074, 2012.
- B. Liu, S. Mahadevan, and J. Liu. Regularized off-policy TD-learning. In *Advances in Neural Information Processing Systems 25*, pages 845–853, 2012.
- H. Maei. *Gradient temporal-difference learning algorithms*. PhD thesis, University of Alberta, 2011.
- S. Mahadevan and B. Liu. Sparse Q-learning with Mirror Descent. In *Proceedings of the Conference on Uncertainty in AI*, 2012.
- S. Mahadevan, B. Liu, P. Thomas, W. Dabney, S. Giguere, N. Jacek, I. Gemp, and J. Liu. Proximal reinforcement learning: A new theory of sequential decision making in primal-dual spaces. *arXiv:1405.6757*, 2014.
- R. Munos and Cs. Szepesvári. Finite time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9:815–857, 2008.
- A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19:1574–1609, 2009.
- R. Parr, C. Painter-Wakefield, L. Li, and M. Littman. Analyzing feature generation for value function approximation. In *Proceedings of the International Conference on Machine Learning*, pages 737–744, 2007.
- LA Prashanth, N. Korda, and R. Munos. Fast LSTD using stochastic approximation: Finite time analysis and application to traffic control. In *Machine Learning and Knowledge Discovery in Databases*, pages 66–81. Springer, 2014.
- Z. Qin and W. Li. Sparse Reinforcement Learning via Convex Optimization. In *Proceedings of the 31st International Conference on Machine Learning*, 2014.
- R. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.
- R. Sutton, C. Szepesvári, and H. Maei. A convergent  $o(n)$  algorithm for off-policy temporal-difference learning with linear function approximation. In *Neural Information Processing Systems*, pages 1609–1616, 2008.
- R. Sutton, H. Maei, D. Precup, S. Bhatnagar, D. Silver, C. Szepesvári, and E. Wiewiora. Fast gradient-descent methods for temporal-difference learning with linear function approximation. In *International Conference on Machine Learning*, pages 993–1000, 2009.
- C. Szepesvári. Algorithms for reinforcement learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 4(1):1–103, 2010.
- M. Tagorti and B. Scherrer. Rate of convergence and error bounds for LSTD ( $\lambda$ ). *arXiv:1405.3229*, 2014.
- H. Yu. Least-squares temporal difference methods: An analysis under general conditions. *SIAM Journal on Control and Optimization*, 50(6):3310–3343, 2012.

---

# Estimating the Partition Function by Discriminance Sampling

---

**Qiang Liu\***

CSAIL, MIT & Computer Science, Dartmouth College  
qliu@cs.dartmouth.edu

**Alexander Ihler**

Information and Computer Science, UC Irvine  
ihler@ics.uci.edu

**Jian Peng\***

Computer Science, UIUC  
jianpeng@illinois.edu

**John Fisher III**

CSAIL, MIT  
fisher@csail.mit.edu

## Abstract

Importance sampling (IS) and its variant, annealed IS (AIS) have been widely used for estimating the partition function in graphical models, such as Markov random fields and deep generative models. However, IS tends to underestimate the partition function and is subject to high variance when the proposal distribution is more peaked than the target distribution. On the other hand, “reverse” versions of IS and AIS tend to overestimate the partition function, and degenerate when the target distribution is more peaked than the proposal distribution. In this work, we present a simple, general method that gives much more reliable and robust estimates than either IS (AIS) or reverse IS (AIS). Our method works by converting the estimation problem into a simple classification problem that discriminates between the samples drawn from the target and the proposal. We give extensive theoretical and empirical justification; in particular, we show that an annealed version of our method significantly outperforms both AIS and reverse AIS as proposed by Burda et al. (2015), which has been the state-of-the-art for likelihood evaluation in deep generative models.

## 1 INTRODUCTION

Probabilistic graphical models, such as Markov random fields, Bayesian networks, and deep generative models provide a powerful set of tools for machine learning (e.g., Lauritzen, 1996, Salakhutdinov and Hinton, 2009). Bayesian analysis utilizing graphical models often involves calculating the partition function, *i.e.* the normalizing constant of the distribution. Unfortunately, such computations are prohibitive (often intractable) for general loopy graphical

models. As such, efficient approximations via variational inference and Monte Carlo methods are of great interest.

Importance sampling (IS) and its variants, such as annealed importance sampling (AIS) (Neal, 2001), are probably the most widely used Monte Carlo methods for estimating the partition function. IS works by drawing samples from a tractable proposal (or reference) distribution  $p_0(x)$ , and estimates the target partition function  $Z = \int f(x)$  by averaging the importance weights  $f(x)/p_0(x)$  across the samples. Unfortunately, the IS estimate often has very high variance if the choice of proposal distribution is very different from the target, especially when the proposal is more peaked or has thinner tails than the target. In addition, in practice IS often underestimates the partition function due to the heavy-tailed nature of the importance weights, leading to overly optimistic likelihood estimates when used for model evaluation (e.g., Burda et al., 2015).

On the other hand, the weighted harmonic mean method (Gelfand and Dey, 1994), which we refer to as a *reverse importance sampling* (RIS), works in an opposite way from IS. It draws samples from the *target distribution*  $p(x) = f(x)/Z$ , and estimates  $Z$  by taking the harmonic mean of the importance weights  $f(x)/p_0(x)$  across these samples. In contrast to IS, reverse IS tends to overestimate the partition function, and gives a high variance when the target distribution is more peaked than the proposal. A reverse version of annealed importance sampling was recently proposed by Burda et al. (2015) to give conservative estimates of test likelihood, in contrast to standard AIS that (like IS) tends to overestimate the likelihood.

Given the opposing properties of IS and RIS, a natural way to improve both is to take the average, or weighted average, of their estimates, in the hope of canceling their individual biases. Unfortunately, the magnitude of bias in IS and RIS can be extremely imbalanced, and it is difficult to decide how much weight each should be given. What is worse, the average would inherit the largest variance between IS and RIS, making even more stringent requirements on the proposal, which should then be neither more peaked nor more flat than the target.

---

\*Both authors contributed equally.

In this work, we study a more efficient and straightforward method for estimating the partition function based on samples from *both* the target distribution and the proposal distribution. The idea is to re-frame estimation of  $Z$  as a simple classification problem that discriminates between the two samples from the target  $p(x)$  and proposal  $p_0(x)$ , respectively. Our method does not have the inherent biases observed in IS and RIS, and is much more robust in that it always has finite variance whenever the proposal and target distributions overlap, a far more mild condition that is easy to satisfy in practice. We provide extensive theoretical and empirical justification for our method. In addition, we show that an annealed importance sampling (AIS) counterpart of our method significantly outperforms AIS, reverse AIS, and their average, which are currently state-of-the-art for model evaluation in deep generative models (Salakhutdinov and Murray, 2008, Burda et al., 2015).

**Outline** The remainder of the paper is organized as follows. We discuss related work in Section 2 and introduce background on IS and RIS in Section 3. Section 4 discusses the proposed method followed by an annealed extension in Section 5. We give experiments in Section 6. The conclusion is provided in Section 7.

## 2 RELATED WORK

The same idea of estimating normalization constants by discriminating between different samples was first proposed independently by Geyer (1991, 1994), although it seems not to be well known in the machine learning community;<sup>1</sup> our work appears to be the first to apply the idea in graphical models, and importantly, propose the annealed version that we show is effective on challenging deep generative models. Another interesting connection can be drawn with a recent noise-matching algorithm (Gutmann and Hyvärinen, 2010) for learning graphical models with intractable partition functions, which is based on a similar idea of discriminating between the observed data (from a *unknown* target distribution) and some artificially generated noise (from the proposal distribution). In fact, our algorithm can be treated as a special noise matching algorithm on a graphical model with only a single unknown parameter  $Z$ . This connection is surprising in part because a naïve likelihood-based or Bayesian inference procedure treating  $Z$  as an unknown parameter fails to work, as discussed in Wasserman (Example 11.10, page 188, 2011) and a thread of related internet discussion (e.g., Wasserman, 2012, and links therein). An intriguing open question is to understand if there exists a principled procedure that turns any *partition function free* learning algorithm, such as Hinton (2002), Lyu (2011), Asuncion et al. (2010), Sohl-Dickstein et al. (2011), into a corresponding *partition function inference* method.

<sup>1</sup> This connection was found by the authors after acceptance.

Related to importance sampling, its use in graphical models almost always require certain variance reduction techniques. Variants of annealed importance sampling based approaches (e.g., Salakhutdinov and Murray, 2008, Theis et al., 2011, Burda et al., 2015, Ma et al., 2013) have been proposed, and are widely used for likelihood evaluation of deep generative models. Other examples of variance reduction methods include adaptive improvement of the proposal (e.g., Cheng and Druzdzal, 2000), and combining with search based methods (e.g., Gogate, 2009, Gogate and Dechter, 2012, 2011) or variational methods (e.g., Wexler and Geiger, 2007). Note that our approach is orthogonal to these developments, and can be combined with them to achieve even better results. In fact, many of our experiments are set up to demonstrate the advantages of combining our method with variational and annealing techniques.

There are also other algorithms that leverage samples from the target distribution. For example, Chib (1995), Chib and Jeliazkov (2001) calculate the marginal likelihood from the output of Gibbs sampling or Metropolis-Hastings. Also related are other generalizations of importance sampling, such as bridge sampling and path sampling (Meng and Wong, 1996, Gelman and Meng, 1998).

## 3 BACKGROUND

Assume we have a distribution  $p(x) = f(x)/Z$ , where  $Z = \int f(x)d\mu(x)$  is the partition function that we are interested in calculating; here the base measure  $\mu(x)$  can be the counting measure for discrete variables, or Lebesgue for continuous variables. We consider Monte Carlo methods for estimating  $Z$ . Two basic methods are the following:

**Importance Sampling (IS)** Assume we have a tractable distribution  $p_0(x)$  which has been properly normalized, that is,  $\int p_0(x)d\mu(x) = 1$ . We draw samples  $\{x_0^1, \dots, x_0^n\}$  from  $p_0(x)$ , and estimate  $Z$  by

$$\hat{Z}_{\text{is}} = \frac{1}{n} \sum_{i=1}^n \frac{f(x_0^i)}{p_0(x_0^i)}.$$

This is an unbiased estimator of  $Z$ , that is,  $\mathbb{E}(\hat{Z}_{\text{is}}) = Z$ , and its mean squared error is known to be

$$n\mathbb{E}\left[\frac{(\hat{Z}_{\text{is}} - Z)^2}{Z^2}\right] = \chi^2(p||p_0) = \int \frac{p^2}{p_0} d\mu(x) - 1, \quad (1)$$

where  $\chi^2(\cdot||\cdot)$  represents the chi-square divergence.

Unfortunately,  $\chi^2(p||p_0)$  is often impractically large, or even infinite, especially when  $p_0(x)$  is more peaked than  $p(x)$ . Additionally, despite the theoretical unbiasedness of  $\hat{Z}$ , it often underestimates  $Z$ . This is due to the distribution of the weights  $f(x)/p_0(x)$  being heavy-tailed with a resulting propensity for outliers. Consequently, the results

using IS may be more properly viewed as a probabilistic lower bound rather than an unbiased estimate (e.g., Burda et al., 2015).

**Reverse Importance Sampling (RIS)** The *weighted harmonic mean method* (Gelfand and Dey, 1994), which we refer to as a reverse importance sampling method, works in an opposite way to importance sampling. It draws samples  $\{x_1^1, \dots, x_1^n\}$  from the *target distribution*  $p(x)$  (e.g., via MCMC when exact sampling is difficult for  $p(x)$ ), and estimates  $Z$  by

$$\hat{Z}_{\text{ris}} = \left[ \frac{1}{n} \sum_{i=1}^n \frac{p_0(x_1^i)}{f(x_1^i)} \right]^{-1}.$$

Note that  $1/\hat{Z}_{\text{ris}}$  can be viewed as a regular importance sampling estimate for  $1/Z$ , justifying  $\hat{Z}_{\text{ris}}$  as a reasonable estimate of  $Z$ . Under regularity conditions (Gelfand and Dey, 1994), the asymptotic MSE of  $\hat{Z}_{\text{ris}}$  (assuming it exists) is

$$n\mathbb{E} \left[ \frac{(\hat{Z}_{\text{ris}} - Z)^2}{Z^2} \right] = \chi^2(p_0||p) = \int \frac{p_0^2}{p} d\mu(x) - 1.$$

The  $\chi^2$ -divergence here has the opposite order as that in (1) for IS, and tends to be large or infinite when  $p(x)$  is more peaked than  $p_0(x)$ . In addition,  $\hat{Z}_{\text{ris}}$  often gives upper bounds on  $Z$  (in contrast to lower bounds by IS), which can be easily seen by viewing  $1/\hat{Z}$  is a regular IS estimate for  $1/Z$ . The special case when  $p_0(x)$  is a uniform distribution is called the *harmonic mean method* (Newton and Raftery, 1994), and sometimes the *Ogata-Tanemura method* (Ogata and Tanemura, 1985).

The fact that IS and RIS give under- and over-estimates respectively suggests the use of their average  $\log \hat{Z}_{\text{avg}} = (\log \hat{Z}_{\text{is}} + \log \hat{Z}_{\text{ris}})/2$  with asymptotic MSE of

$$n\mathbb{E} \left[ \frac{(\hat{Z}_{\text{avg}} - Z)^2}{Z^2} \right] = \frac{1}{4} (\chi^2(p||p_0) + \chi^2(p_0||p)). \quad (2)$$

Unfortunately, this is large whenever one of  $\chi^2(p||p_0)$  or  $\chi^2(p_0||p)$  is large and thus imposes more stringent constraints on  $p_0$  such that (2) is finite, *i.e.* it can neither be too peaked nor too flat compared to the target distribution. This can be significant even for simple distributions as in the following example.

**Example 1.** Consider normal distributions  $p(x) = \mathcal{N}(x; 0, \sigma^2)$  and  $p_0(x) = \mathcal{N}(x; 0, \sigma_0^2)$ . One can show that  $\text{var}(\hat{Z}_{\text{is}}) = +\infty$  if  $\sigma_0 \leq \sigma/\sqrt{2}$  ( $p_0$  is much more peaked than  $p$ ). Conversely,  $\text{var}(\hat{Z}_{\text{ris}}) = +\infty$  if  $\sigma \leq \sigma_0/\sqrt{2}$  ( $p$  is much more peaked than  $p_0$ ). Therefore, their average  $Z_{\text{avg}}$  has finite variance only when  $\sigma/\sqrt{2} \leq \sigma_0 \leq \sqrt{2}\sigma$ .

More advanced combinations of IS and RIS can be obtained by estimating their variances, and taking weighted

averages, or selecting the better one according to their variance. Unfortunately, the variance estimates themselves are unreliable, often over- or under-estimated, making these methods ineffective. We explore and compare several of these options in our experiments; see Section 6 for details.

## 4 DISCRIMINANCE SAMPLING

Here we propose a new estimator of  $Z$ , termed *discriminance sampling* (evoking *importance* and *discriminative*), based on both  $\{x_1^i\} \sim p(x)$  and  $\{x_0^i\} \sim p_0(x)$  jointly. The idea is to reframe estimation of  $Z$  as a classification problem between  $\{x_1^i\}$  and  $\{x_0^i\}$ . To start, we assign a binary label  $y_1^i = 1$  for each  $x_1^i \sim p(x)$ , and correspondingly  $y_0^i = 0$  for each  $x_0^i \sim p_0(x)$ . Putting these samples together we get  $\{x^i\} = \{x_1^i\} \cup \{x_0^i\}$  and  $\{y^i\} = \{y_1^i\} \cup \{y_0^i\}$ . In this way, the conditional distribution of  $y^i$  given  $x^i$  is

$$p(y^i = 0 | x^i) = \frac{p_0(x^i)}{f(x^i)/Z + p_0(x^i)},$$

where  $Z$  can be treated as an unknown parameter. This motivates a parameter estimation procedure where we consider a family of conditional probabilities  $p(y = 1|x; c) = \frac{p_0(x)}{f(x)/c + p_0(x)}$ , indexed by a parameter  $c$ , and estimate  $c$  by maximizing the conditional likelihood:

$$\hat{Z}_{\text{dis}} = \arg \max_{c: c \geq 0} \sum_{i=1}^{2n} \log \frac{y^i f(x^i)/c + (1 - y^i) p_0(x^i)}{f(x^i)/c + p_0(x^i)}.$$

Calculating the zero-gradient equation of the objective function, we see that the optimal  $c$  should satisfy the following ratio matching condition:

$$\frac{1}{2n} \sum_{i=1}^{2n} \frac{p_0(x^i)}{f(x^i)/c + p_0(x^i)} = \frac{1}{2}, \quad (3)$$

that is, the proportion of  $y = 0$  (and  $y = 1$ ) predicted by the model should equal  $1/2$ , matching the label proportions in the data. Because the LHS of (3) is an increasing function on  $c$ , Eq. (3) yields a unique solution unless  $p(x^i)p_0(x^i) = 0$  for all  $i$ , that is, when  $p(x)$  and  $p_0(x)$  do not overlap. In practice, we can solve (3) efficiently using root finding algorithms such as the `fzero` function in MATLAB.

**Proposition 1.** Assume  $\{x_1^i\}_{i=1}^n$  and  $\{x_0^i\}_{i=1}^n$  are i.i.d. samples from  $p(x)$  and  $p_0(x)$ , respectively. Let  $e_1 = \sqrt{2n}(\hat{Z}_{\text{dis}}/Z - 1)$  and  $e_2 = \sqrt{2n}(\log \hat{Z}_{\text{dis}} - \log Z)$ . Define

$$\gamma = \mathbb{E}[\text{var}(y|x)] = \frac{1}{2} \int \frac{p(x)p_0(x)}{p(x) + p_0(x)} d\mu(x), \quad (4)$$

then if  $\gamma \neq 0$ , we have  $\hat{Z}_{\text{dis}} \xrightarrow{a.s.} Z$  as  $n \rightarrow \infty$ , and  $e_1$  and  $e_2$  have a normal distribution  $\mathcal{N}(0, (\frac{1}{4} - \gamma)/\gamma^2)$ .

*Proof.* Apply the standard asymptotic result in DasGupta (Theorem 17.2, Page 264, 2008); the condition  $\gamma \neq 0$  guarantees (3) has a unique solution as  $n \rightarrow \infty$ . Note that  $e_1$  and  $e_2$  are asymptotically equivalent because  $\log \epsilon \approx \epsilon - 1$  for  $\epsilon \approx 1$ .  $\square$

*Remarks.* (i) Eq (4) above relates the accuracy of  $\hat{Z}_{\text{dis}}$  with the variance of the label  $y$  given  $x$ , which is a measure of distinguishability between the two samples  $\{x_1^i\}$  and  $\{x_0^i\}$ . Ideally, we want to choose  $p_0$  so that it is hard to distinguish between  $\{x_1^i\}$  and  $\{x_0^i\}$ ; when  $p_0 = p$ ,  $\hat{Z}_{\text{dis}}$  equals  $Z$  exactly.

(ii) The variance of  $\hat{Z}_{\text{dis}}$  is infinite only if  $\gamma = 0$ , that is,  $\int \frac{pp_0}{p+p_0} = 0$ ; this is possible only if  $p(x)$  and  $p_0(x)$  do not overlap, that is,  $p(x)p_0(x) = 0$  almost everywhere. Note that this is a much milder condition compared to that for IS, RIS and their average, since in practice it is usually easy to choose a  $p_0(x)$  that shares some support with  $p(x)$ .

**Example 2.** To continue with Example 1, the MSE of  $\hat{Z}_{\text{dis}}$  is finite for any  $\sigma_0 > 0$  and  $\sigma > 0$ , making it far more robust than IS or reverse IS, which have finite MSE only when  $\sigma_0 > \sigma/\sqrt{2}$  and  $\sigma_0 < \sqrt{2}\sigma$ , respectively.

## 5 ANNEALED DISCRIMINANCE SAMPLING

One advantage of our method is that it can be naturally extended to cases when we have more than two distributions, in which case it is straightforward to frame a corresponding multinomial classification problem. In this section, we consider an improvement to our method by introducing a set of auxiliary distributions that serve as intermediate points between the target and reference distributions. This extension is analogous to annealed importance sampling (AIS) (Neal, 2001, Salakhutdinov and Murray, 2008) and reverse AIS (Burda et al., 2015), but with significantly better performance.

Both AIS and reverse AIS are based on a set of distributions  $\{p_k = f_k(x)/Z_k : k = 0, \dots, m\}$  where  $p_0$  is the normalized reference distribution ( $Z_0 = 1$ ) and  $p_m(x) = f(x)/Z$  is the target distribution; the other distributions serve as “intermediate points” between  $p_0$  and  $p$ . A typical choice of the distributions is  $f_k(x) = f(x)^{k/m} f_0(x)^{1-k/m}$ , where  $k$  can be interpreted as a temperature parameter that anneals between  $p$  and  $p_0$ .

Now assume we draw  $m$  sets of samples  $\{x_k^i\}_{i=1}^n \sim p_k(x)$ ,  $k = 0, \dots, m$ . Similar to Section 4, we assign each  $x_k^i$  with a label  $y_k^i = k$ , resulting a conditional likelihood of

$$p(y^i = k | x^i) = \frac{f_k(x^i)/Z_k}{\sum_{k=0}^m f_k(x^i)/Z_k}.$$

We then treat  $\{Z_k : k = 1, \dots, m\}$  as a set of unknown parameters, and estimate them by performing maximum con-

---

### Algorithm 1 Annealed Discriminace Sampling (Sequential Binary Version)

---

Draw  $x_0^i \sim p_0(x)$ . Set  $w_0^i = 1$  and  $Z_0 = 1$ .

**for**  $k = 1$  to  $m$  **do**

    Generate weighted sample  $\{x_k^i, w_k^i\}_{i=1}^n$  by the AIS update in (8).

    update  $\hat{Z}_k = Z_{k-1} \hat{r}_k$ , where  $\hat{r}_k$  maximizes the weighted conditional likelihood (9).

**end for**

**Return:**  $\hat{Z}_m$  is an estimate of the partition function  $Z$ .

---

ditional likelihood:

$$\{\hat{Z}_k\}_{k=0}^m = \arg \max_{c_0: c_0=Z_0} \sum_{k=0}^m \sum_{i=1}^n \log \frac{f_k(x_k^i)/c_k}{\sum_{k=0}^m f_k(x_k^i)/c_k}, \quad (5)$$

where  $c_0$  is fixed to its known value ( $Z_0 = 1$ ). Similar to the binary case, it is easy to show that  $\{\hat{Z}_k\}$  forms a consistent estimation of  $\{Z_k\}$  (although we are only interested in  $\hat{Z}_m$ ).

Note that (5) is a convex optimization w.r.t.  $\{\log c_k\}$ , and can be solved efficiently. A further simplification of (5) is to construct and combine a sequence of binary classifications between the  $(p_k, p_{k+1})$  pairs, instead of the joint multinomial classification. To be specific, we sequentially estimate the ratio  $r_{k+1} = Z_{k+1}/Z_k$  between  $p_k$  and  $p_{k+1}$  by discriminating between  $\{x_k^i\}$  and  $\{x_{k+1}^i\}$ :

$$\hat{r}_{k+1} = \arg \max_{\substack{c_{k+1} > 0 \\ c_k = 1}} \sum_{k'=k}^{k+1} \sum_{i=1}^n \log \frac{f_{k'}(x_{k'}^i)/c_{k'}}{\sum_{\ell=k}^{k+1} f_{\ell}(x_{k'}^i)/c_{\ell}} \quad (6)$$

and estimate  $Z = Z_m$  by chaining the ratios together:

$$\log \hat{Z} = \sum_{k=1}^m \log \hat{r}_k \approx \sum_{k=1}^m \log r_k = \log Z. \quad (7)$$

Interestingly, we find that such sequential binary classification works as well as the joint multinomial classification in our experiments, possibly because the neighboring  $\{p_k, p_{k+1}\}$  are close to each other and provide more accurate estimates of their ratios.

**Practical Implementation** In practice, it is expensive to sample from all  $p_k(x) = f_k(x)/Z_k$  at each temperature independently, especially when the number of temperatures is large. Instead, we use AIS (Neal, 2001) to sequentially generate *importance weighted samples* for each  $p_k(x)$ : we start with  $x_0^i \sim p_0(x)$  and set  $w_0^i = 1$ , and sequentially update the samples using Markov chain transitions and adjust the weights accordingly:

$$x_k^i \sim T_k(\cdot | x_{k-1}^i), \quad w_k^i = w_{k-1}^i \frac{f_k(x_{k-1}^i)}{f_{k-1}(x_{k-1}^i)}, \quad (8)$$

---

**Algorithm 2** Annealed Discriminace Sampling (Multinomial Version)

---

1. Use AIS to generate  $\{x_k^i, w_k^i\}_{i=1}^n$  for  $\forall 0 \leq k \leq m$ .
2. Estimate  $\hat{Z}_k$  by maximizing

$$\{\hat{Z}_k\}_{k=0}^m = \arg \max_{c>0: c_0=Z_0} \sum_{k=0}^m \sum_{i=1}^n \tilde{w}_k^i \log \frac{f_k(x_k^i)/c_k}{\sum_{k=0}^m f_k(x_k^i)/c_k},$$

where  $\tilde{w}_k^i = w_k^i / \sum_i w_k^i$  are the normalized weights.

**Return:**  $\hat{Z}_m$  is an estimate of the partition function  $Z$ .

---

for  $\forall 1 \leq k \leq m$ , where  $T_k(\cdot)$  is a Gibbs or Metropolis-Hastings transition kernel of  $p_k$ . By the augmented variable space argument of Neal (2001), we can show the weighted sample  $(x_k^i, w_k^i)_{i=1}^n$  follows  $p_k(x)$  in the sense that

$$\mathbb{E}(w_k^i h(x_k^i)) = \text{const} \cdot \mathbb{E}_{x \sim q_k}(h(x))$$

for any  $0 \leq k \leq m$  and integrable function  $h(x)$ ; this allows us to estimate the partition functions  $Z_k$  using weighted versions of the multinomial (5) or sequential binary (6) classifications based on the weighted samples  $(x_k^i, w_k^i)$ . For example, we can estimate the ratio  $r_{k+1} = Z_{k+1}/Z_k$  between  $p_{k+1}$  and  $p_k$  by maximizing a weighted version of the conditional likelihood in (6),

$$\hat{r}_{k+1} = \arg \max_{\substack{c_{k+1}>0 \\ c_k=1}} \sum_{k'=k}^{k+1} \sum_{i=1}^n \tilde{w}_{k'}^i \log \frac{f_{k'}(x_{k'}^i)/c_{k'}}{\sum_{\ell=k}^{k+1} f_{\ell}(x_{k'}^i)/c_{\ell}}, \quad (9)$$

where  $\tilde{w}_k^i = w_k^i / \sum_{i=1}^n w_k^i$  are the normalized weights under each temperature  $k$ . See Algorithm 1 for the full algorithm of the sequential binary version of our method; the corresponding multinomial version is shown in Algorithm 2. Note that both Algorithm 1 and 2 can recycle the samples and weights generated by AIS and can be implemented conveniently based on AIS. Alternatively, we can also base our estimator on other methods that draw samples jointly from different temperatures, such as simulated tempering and parallel tempering (see Liu, 2008, and references therein).

## 6 EXPERIMENTS

We present experimental results on a toy Gaussian example, pairwise Markov random fields ( $10 \times 10$  grids), and deep generative models trained on real world data. Our contributions are threefold: (1) We demonstrate the advantage of our method compared to IS, reverse IS and their combinations, and show that our method yields significantly smaller bias and variance across all our experiments. (2) We illustrate the benefits of combining deterministic variational methods and the Monte Carlo based methods discussed in this paper; we show that Monte Carlo methods can provide tighter (but “probabilistic”) bounds than

deterministic variational methods, and can be further improved by using variational methods to provide better reference distributions  $p_0$ . (3) We test the annealed version of our algorithm in real-world deep learning models, including restricted Boltzmann machines (RBM) and deep Boltzmann machines (DBM), and show that it significantly outperforms the state-of-the-art AIS and reverse AIS methods (Burda et al., 2015).

**Setting** We compare our algorithm with IS, RIS and three different methods that combine IS and RIS in hopes of off-setting their relative biases:

(1) *Naive Averaging*:

$$\log \hat{Z}_{\text{avg}} = (\log \hat{Z}_{\text{is}} + \log \hat{Z}_{\text{ris}})/2.$$

Here the average is taken on the log domain; note that averaging in the  $Z$  domain, i.e.,  $(\hat{Z}_{\text{is}} + \hat{Z}_{\text{ris}})/2$  does not make sense since we almost always have  $\hat{Z}_{\text{is}} \ll \hat{Z}_{\text{ris}}$ , and the result will be dominated by  $\hat{Z}_{\text{ris}}$ .

(2) *Weighted Averaging*:

$$\log \hat{Z}_{\text{w}} = (\hat{v}_{\text{is}}^{-1} \log \hat{Z}_{\text{is}} + \hat{v}_{\text{ris}}^{-1} \log \hat{Z}_{\text{ris}}) / (\hat{v}_{\text{is}}^{-1} + \hat{v}_{\text{ris}}^{-1}),$$

where  $\hat{v}_{\text{is}}, \hat{v}_{\text{ris}}$  are empirical estimates of  $\text{var}(\log \hat{Z}_{\text{is}})$  and  $\text{var}(\log \hat{Z}_{\text{ris}})$ ,

$$\hat{v}_{\text{is}} = \widehat{\text{var}}\left(\left\{\frac{f(x_0^i)}{p_0(x_0^i)}\right\}\right) / \hat{Z}_{\text{is}}^2,$$

$$\hat{v}_{\text{ris}} = \widehat{\text{var}}\left(\left\{\frac{p_0(x_1^i)}{f(x_1^i)}\right\}\right) \hat{Z}_{\text{ris}}^2,$$

where  $\widehat{\text{var}}(\cdot)$  represents the empirical variance estimate. Note that the weights defined above should minimize the variance of the combination if the variance estimates are accurate. Unfortunately, variance estimates for the weighted averaging approach are typically unreliable and have the same under- / over-estimation problem as IS and RIS, causing the weighted average to perform poorly.

(3) *Weighted Selection*:

$$\log \hat{Z}_{\text{s}} = [\hat{v}_{\text{is}} < \hat{v}_{\text{ris}}] \log \hat{Z}_{\text{is}} + [\hat{v}_{\text{ris}} < \hat{v}_{\text{is}}] \log \hat{Z}_{\text{ris}},$$

where we select the estimator with smaller estimated variance; here  $[\cdot]$  is the indicator function.

Note that IS (resp. RIS) uses only  $\{x_0^i\}_{i=1}^n \sim p_0(x)$  (resp.  $\{x_1^i\}_{i=1}^n \sim p(x)$ ), while our method uses both  $\{x_0^i\}$  and  $\{x_1^i\}$ . To make a conservative comparison, we use *only the first half of the samples*  $\{x_1^i\}_{i=1}^{n/2}$  and  $\{x_0^i\}_{i=1}^{n/2}$  in our method. However, we do not halve the samples when calculating the averages  $\hat{Z}_{\text{avg}}, \hat{Z}_{\text{w}}$  and  $\hat{Z}_{\text{s}}$ , allowing them to use *exactly twice the information* as our method. Despite these unfavorable conditions, our method still consistently outperforms IS, RIS and all the averaging based methods.



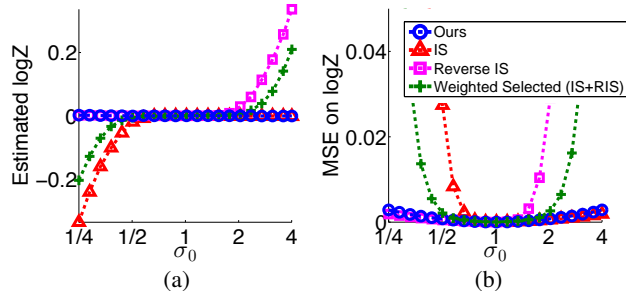


Figure 1: Gaussian toy example. The estimated values (a) and mean square errors (b) on  $\log Z$  by different methods (the true value is  $\log Z = 0$ ). IS performs poorly when  $\sigma_0$  is small ( $p_0$  is too peaked), while reverse IS is poor when  $\sigma_0$  is large ( $p$  is too peaked). Our method performs much better and is robust for all values of  $\sigma_0$ . The result is averaged over 1000 random trials.

**Gaussian Toy Example** As in Example 1, we consider  $p(x) = \mathcal{N}(x; 0, \sigma^2)$  with fixed  $\sigma = 1$  and  $p_0(x) = \mathcal{N}(x; 0, \sigma_0^2)$  with different values of  $\sigma_0$ . We are interested in calculating the normalization constant of  $p(x)$ , which is trivially  $Z = 1$  in this case. We use  $n = 1000$  samples from both the target  $p(x)$  and reference  $p_0(x)$ .

Figure 1 reports the bias and MSE on  $\log Z$  as returned by our methods, IS, reverse IS, and the weighted selection  $\log \hat{Z}_s$  (the naïve average  $\log \hat{Z}_{\text{avg}}$  and weighted average  $\log \hat{Z}_w$  are worse than  $\log \hat{Z}_s$  and not shown in the figure for clarity). We find that our method significantly outperforms all the other algorithms, despite using fewer samples than the averaging based methods.

The performance of IS and reverse IS in Figure 1(a) is consistent with the theoretical analysis: IS tends to give a lower bound, and degenerates quickly when  $\sigma_0$  is small ( $p_0$  is more peaked than  $p$ ), while reverse IS gives an upper bound, and degenerates when  $\sigma_0$  is large ( $p$  is more peaked than  $p_0$ ). In this case, it is interesting to see that the performances of IS and reverse IS are extremely imbalanced and anti-correlated (whenever IS performs well, RIS performs poorly, and vice versa), which explains why weighted selection is better than naïve averaging in this case.

**MRF on  $10 \times 10$  Grid** We consider Markov random fields (MRFs) on a  $10 \times 10$  grid

$$p(x) = \frac{1}{Z} \exp \left( \sum_{ij} \theta_{ij}(x_i, x_j) + \sum_i \theta_i(x_i) \right),$$

where  $x_i \in \{0, 1\}$ . We generate each  $\theta_i(k)$  randomly by  $\mathcal{N}(0, \sigma_s^2)$ , with fixed  $\sigma_s = 0.1$  and each  $\theta_{ij}(k, l)$  from  $\mathcal{N}(0, \sigma_p^2)$ , where  $\sigma_p$  characterizes the interaction strength in the MRF. We also explore different choices of reference distribution  $p_0$  for the MRF, including

(1) *Uniform distribution* as shown in Figure 2(a).

(2) *Mean field* approximation as shown in Figure 2(b).

(3) *Mixture of trees* constructed from the reparameterization obtained from tree reweighted belief propagation (TRBP) (Wainwright et al., 2005) (Figure 2(c)). The edge appearance probabilities in TRBP are set by assigning uniform weights to a random set of spanning trees.

The samples from  $p_0$  are drawn exactly, while those from  $p$  are drawn using Gibbs sampling with 500 burn-in steps. We use  $n = 1000$  samples from each distribution in all cases, and average the results over 500 random trials.

From Figure 2, we find our method significantly outperforms IS and reverse IS, and all the versions of their combinations under all three choices of  $p_0$ . The deterministic bounds returned by TRBP and MF are shown in Figure 2(a), and are significantly looser than the sampling based bounds in these cases (which, however, provides probabilistic, instead of deterministic bound guarantees as the variational methods). We note that the proposal produced by MF is only as good as the uniform proposal. On the other hand, the  $p_0$  produced by a mixture of TRBP trees gives significantly better results (note that the y-axes are not on the same scale). This result demonstrates the potential of combining variational methods and sampling methods, with carefully designed choices for  $p_0$  and estimation methods (such as our method).

Interestingly, we find the performance of IS and reverse IS are relatively balanced in the MRF examples, making the naïve average of IS and reverse IS outperform both the weighted average and weighted selection. This is in contrast to the Gaussian toy example, where IS and reverse IS are extremely imbalanced. Unfortunately, there is no general method to tell whether IS and reverse IS will be balanced or not beforehand.

**Deep Generative Models** We compare our annealed discriminant sampling (ADS) with the AIS and reverse AIS estimator (RAISE) as introduced in Burda et al. (2015) for partition function estimation in deep generative models, including a restricted Boltzmann machine (RBM) and a deep Boltzmann machine (DBM). We implement AIS and RAISE following Algorithm 1 and Algorithm 3<sup>2</sup> in Burda et al. (2015), respectively. We then take the samples and weights generated by AIS and run our sequential binary ADS in Algorithm 1 and multinomial ADS in Algorithm 2. In principle, we can also reuse the same samples generated by AIS to construct a version of a reverse AIS estimator. Unfortunately, we find this works poorly in practice, and it seems to be important to follow Algorithm 3 in Burda et al. (2015) to generate new samples specifically for the

<sup>2</sup> Algorithm 3 of Burda et al. (2015) was designed for calculating the testing likelihood; we adopt it for calculating the partition function by replacing its conditional kernel  $\tilde{T}_k^{(\text{v}_{\text{test}})}(\cdot | h_{k-1})$  in the forward step with the unconditional kernel  $\tilde{T}_k(\cdot | x'_{k-1})$ .

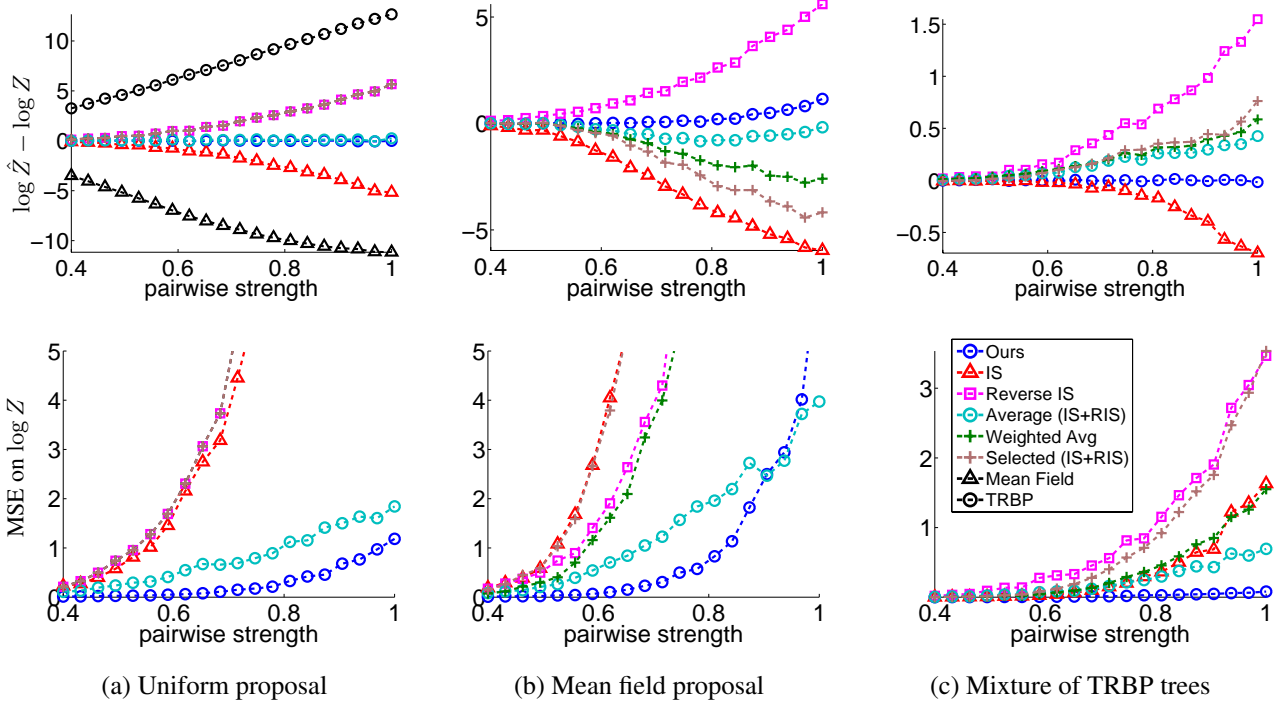


Figure 2: MRFs on a  $10 \times 10$  grid. The three columns represent the results when using different reference distributions  $p_0$ , including the uniform distribution (a), mean field approximation (b) and a mixture of trees constructed from TRBP (c). Our algorithm consistently performs best. Note that the comparison is again in favor of the averaging methods since they use twice as many samples as our method.

reverse AIS estimates. Note that implemented in this way, the average of AIS and RAISE uses twice the number of samples as our method. In addition, we emphasize that the RAISE as proposed in Algorithm 3 in Burda et al. (2015) includes both a forward and backward sampling step, requiring twice the computational cost of AIS. In contrast, our method has roughly the same time complexity as AIS, because the cost of the discriminance analysis step in our method, especially the sequential binary version, is negligible compared to the sample generation steps of AIS as used in Algorithm 1 in Burda et al. (2015).

In both our experiments for RBM and DBM and for all the annealing-based algorithms, we use  $2^1 \sim 2^{10}$  linearly spaced intermediate temperatures (or distributions) and  $n = 1000$  samples (corresponding to 1000 separate MCMC chains in AIS). The reference distribution  $p_0$  is taken to be the *data base rate* (DBR) distribution as suggested by Salakhutdinov and Hinton (2009), which is constructed based the marginal statistics of the image dataset. In all cases, we repeat the estimates 10 times and report the average bias and MSE results.

To obtain the true partition function of both RBM and DBM, we calculate the average of AIS and RAISE with an extremely large number (in our case, 100,000) of temperatures, until their estimates coincide to within 0.1 nats, i.e.,

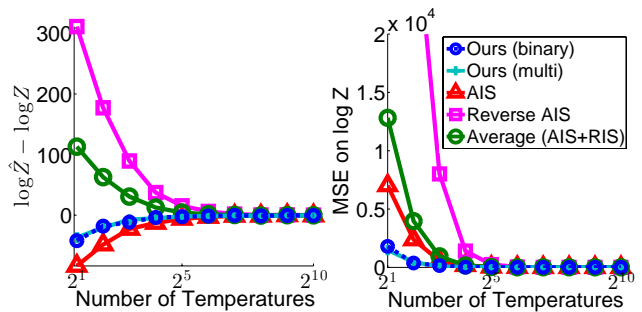


Figure 3: Estimates of log-partition function of a restricted Boltzmann machine trained on MNIST with different numbers of intermediate temperatures. While all methods converge to the same value, our method significantly outperforms other methods when the intermediate temperatures are few.

$|\log \hat{Z}_{\text{is}} - \log \hat{Z}_{\text{ris}}| \leq 0.1$ . This gives a high confidence estimate of the true partition function, since AIS and RAISE are probabilistic lower and upper bounds, respectively.

We first consider a restricted Boltzmann machine (RBM) with 500 hidden nodes trained on MNIST using contrastive divergence with 25 steps. Figure 3 shows the results of different algorithms. When there are many intermediate

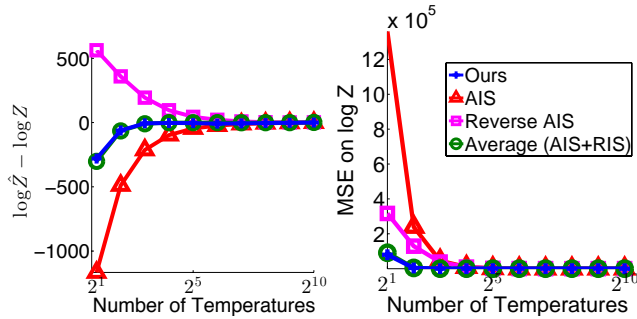


Figure 4: Estimates of log-partition function of a 784-500-1000 deep Boltzmann machine trained on MNIST with different numbers of intermediate temperatures. While all methods converge to the same value, our method outperforms both AIS and RAISE when the intermediate temperatures are few. Our multinomial ADS performs the same as our binary version, and is omitted in the figure for clarity.

temperatures, all algorithms give accurate estimates. When there are fewer intermediate temperatures, our ADS is able to compute significantly more accurate estimates than AIS and RAISE, or even their average. In addition, we find that the binary and multinomial versions of our ADS algorithm work similarly (almost identically) in all our experiments.

We then experiment on a more complex deep Boltzmann machine, trained with a 784-500-1000 structure on MNIST closely following the procedure in Salakhutdinov and Hinton (2009): we initially train the first layer RBM for 100 epochs, then the second layer with 200 epochs and then fine-tune the two layers jointly for 300 epochs. The results of the different algorithms are shown in Figure 4. Similarly to the results in the RBM experiment, ADS significantly outperforms both AIS and RAISE when the number of temperatures is small. In this case, we find that the average of AIS and RAISE is quite accurate for this DBM model, almost as good as ADS. However, our algorithm still gives significant advantages in practice: again, we use only half the number of samples that are used by the average of AIS and RAISE and have only 1/3 of the total computational cost (because RAISE is twice as expensive as AIS or our ADS method, as discussed previously).

## 7 CONCLUSION

In this paper, we introduced *discriminance* sampling, a novel and efficient method for estimating log-partition functions of probabilistic distributions. Using samples drawn from both the target and proposal distributions, we formulated the estimation problem into a discriminant analysis problem that classifies samples into their corresponding distributions. Our approach does not under- / overestimate the true values like IS and reverse IS, and places much less stringent requirements on the proposal distribu-

tions. In addition, we also extend our method to define annealed discriminance sampling (ADS) and demonstrate that ADS significantly outperform AIS, reverse AIS, and performs as well or better than their average, which are currently state-of-the-art methods for model evaluation in deep generative models.

**Acknowledgement** This work is supported in part by VITALITE, which receives support from Army Research Office (ARO) Multidisciplinary Research Initiative (MURI) program (Award number W911NF-11-1-0391); NSF grants IIS-1065618 and IIS-1254071; and by the United States Air Force under Contract No. FA8750-14-C-0011 under the DARPA PPAML program.

## References

Asuncion, A., Liu, Q., Ihler, A., and Smyth, P. (2010). Learning with blocks: Composite likelihood and contrastive divergence. In *AISTATS*.

Burda, Y., Grosse, R., and Salakhutdinov, R. (2015). Accurate and conservative estimates of MRF log-likelihood using reverse annealing. In *AISTATS*.

Cheng, J. and Druzdzel, M. (2000). AIS-BN: An adaptive importance sampling algorithm for evidential reasoning in large Bayesian networks. *Journal of Artificial Intelligence Research*.

Chib, S. (1995). Marginal likelihood from the Gibbs output. *Journal of the American Statistical Association*, 90(432):1313–1321.

Chib, S. and Jeliazkov, I. (2001). Marginal likelihood from the Metropolis–Hastings output. *Journal of the American Statistical Association*, 96(453):270–281.

DasGupta, A. (2008). *Asymptotic theory of statistics and probability*. Springer Science & Business Media.

Gelfand, A. and Dey, D. (1994). Bayesian model choice: asymptotics and exact calculations. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 501–514.

Gelman, A. and Meng, X.-L. (1998). Simulating normalizing constants: From importance sampling to bridge sampling to path sampling. *Statistical science*, pages 163–185.

Geyer, C. (1991). Reweighting monte carlo mixtures. Technical Report 568, School of Statistics, University of Minnesota.

Geyer, C. (1994). Estimating normalizing constants and reweighting mixtures in Markov chain Monte Carlo. Technical Report 568, School of Statistics, University of Minnesota.

Gogate, V. (2009). *Sampling Algorithms for Probabilistic Graphical Models with Determinism*. PhD thesis, University of California, Irvine.

- Gogate, V. and Dechter, R. (2011). SampleSearch: Importance sampling in presence of determinism. *Artificial Intelligence*, 175(2):694–729.
- Gogate, V. and Dechter, R. (2012). Importance sampling-based estimation over AND/OR search spaces for graphical models. *Artificial Intelligence*, 184–185(0):38 – 77.
- Gutmann, M. and Hyvärinen, A. (2010). Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *ICML*.
- Hinton, G. (2002). Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800.
- Lauritzen, S. (1996). *Graphical models*. Oxford University Press.
- Liu, J. (2008). *Monte Carlo strategies in scientific computing*. Springer Science & Business Media.
- Lyu, S. (2011). Unifying non-maximum likelihood learning objectives with minimum KL contraction. In *NIPS*.
- Ma, J., Peng, J., Wang, S., and Xu, J. (2013). Estimating the partition function of graphical models using Langevin importance sampling. In *AISTATS*.
- Meng, X. and Wong, W. (1996). Simulating ratios of normalizing constants via a simple identity: a theoretical exploration. *Statistica Sinica*, 6(4):831–860.
- Neal, R. (2001). Annealed importance sampling. *Statistics and Computing*, 11(2):125–139.
- Newton, M. and Raftery, A. (1994). Approximate Bayesian inference with the weighted likelihood bootstrap. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 3–48.
- Ogata, Y. and Tanemura, M. (1985). Estimation of interaction potentials of marked spatial point patterns through the maximum likelihood method. *Biometrics*, pages 421–433.
- Salakhutdinov, R. and Hinton, G. (2009). Deep Boltzmann machines. In *AISTATS*.
- Salakhutdinov, R. and Murray, I. (2008). On the quantitative analysis of deep belief networks. In *ICML*.
- Sohl-Dickstein, J., Battaglino, P., and DeWeese, M. (2011). Minimum probability flow learning. In *ICML*.
- Theis, L., Gerwinn, S., Sinz, F., and Bethge, M. (2011). In all likelihood, deep belief is not enough. *The Journal of Machine Learning Research*, 12:3071–3096.
- Wainwright, M., Jaakkola, T., and Willsky, A. (2005). A new class of upper bounds on the log partition function. *Information Theory, IEEE Transactions on*, 51(7):2313–2335.
- Wasserman, L. (2011). *All of statistics*. Springer Science & Business Media.
- Wasserman, L. (2012). The normalizing constant paradox. <https://normaldeviate.wordpress.com/2012/10/05/the-normalizing-constant-paradox/>.
- Wexler, Y. and Geiger, D. (2007). Importance sampling via variational optimization. In *UAI*.

---

# A Finite Population Likelihood Ratio Test of the Sharp Null Hypothesis for Compliers

---

**Wen Wei Loh**

Department of Statistics  
University of Washington  
wloh@u.washington.edu

**Thomas S. Richardson**

Department of Statistics  
University of Washington  
thomasr@u.washington.edu

## Abstract

In a randomized experiment with noncompliance, scientific interest is often in testing whether the treatment exposure  $X$  has an effect on the final outcome  $Y$ . We propose a finite-population significance test of the sharp null hypothesis that  $X$  has no effect on  $Y$ , within the principal stratum of Compliers, using a generalized likelihood ratio test. We present a new algorithm that solves the corresponding integer programs.

## 1 INTRODUCTION

Randomized experiments are often employed in order to determine whether a treatment  $X$  has a causal effect on an outcome  $Y$ . For example, individuals may be randomly assigned to either an active treatment group ( $X = 1$ ) or to the placebo or control group ( $X = 0$ ).

This problem may be formulated in terms of potential outcomes. Denote  $Y(x = 1)$  as the outcome that the patient would have if assigned to the treatment arm, while  $Y(x = 0)$  is the outcome that would arise under placebo. The absence of an effect of  $X$  on  $Y$  when the sharp causal null holds is formalized by  $Y(x = 1) = Y(x = 0)$ , such that every individual in the finite population has the same outcome regardless of the treatment group  $X$  to which they were assigned [19].

Randomization of treatment implies that  $X \perp\!\!\!\perp \{Y(x = 1), Y(x = 0)\}$ . Under the sharp causal null, this then implies  $X \perp\!\!\!\perp Y$ . Hence testing this latter independence may thus be seen as a test of the sharp causal null. For the case of binary outcomes  $Y$ , we may use Fisher's exact test [5], see for example [16, pg. 308].

A key feature of the potential outcome framework is that the set of individuals in the population and the values of their potential outcomes are regarded as fixed. Differences between results over hypothetical replications arise *only*

due to different random assignments of this fixed set of individuals to treatment or control.

However, often we are interested in the effect of a treatment  $X$  that was not randomized. In this paper we consider the circumstance where, although  $X$  is not randomized, there is another variable  $Z$ , called an 'instrument' that is randomized, and influences  $X$ , but does not influence  $Y$  directly; see Figure 1.

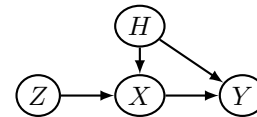


Figure 1: Graphical Representation of the Instrumental Variable Model, where  $H$  are unobserved confounding variables.

A common example of this circumstance is a randomized study with 'noncompliance'. In this context  $Z$  represents the assigned treatment, while  $X$  is the treatment that the patient actually receives.  $X$  and  $Z$  may differ owing to noncompliance.

Randomized experiments with treatment 'noncompliance' arise in many situations. For example, in a randomized psychology experiment, whether or not participants adhere to their assigned treatment depends on their personalities and the type of manipulation (treatment). Patients in a randomized clinical trial may choose not to take their prescribed treatment, possibly due to side-effects. In studies where a randomly selected subset of subjects are offered an incentive to avail themselves of a treatment, or 'encouragement' studies, the inducement may be sufficient for some but not for others.

For each of these randomized experiments, every unit now has a treatment actually received ( $X$ ) that was *not* randomized, following an *assigned* treatment ( $Z$ ) that was randomized. We will make the assumption that  $Z$  has no (direct) effect on  $Y$  except through  $X$ , sometimes termed an 'exclusion restriction'.

In such studies with noncompliance with a binary treatment, Angrist et al. [1] and Imbens and Rubin [8] among others, propose to find the effect of treatment on the subset of individuals who would conform with the assigned treatment regardless of the arm to which they are assigned. Sommer and Zeger [18] describe this subgroup of individuals as ‘Compliers’: individuals who would take the treatment only if assigned to do so and would not if assigned not to do so. Balke and Pearl [2] used a symbolic linear program to derive the bounds for counterfactual probabilities, and the average causal effect of  $X$  on  $Y$ . Rubin [17] uses randomization-based posterior-predictive p-values to test a null treatment effect; Imbens and Rosenbaum [7] use randomization-based inference to obtain valid confidence intervals for the treatment effect under an additive structural model even when the instrument is ‘weak’.

In this paper we address the problem of testing the sharp null hypothesis of no effect of  $X$  on  $Y$  for ‘Compliers’, the subpopulation or principal stratum where  $X(z=0)=0$  and  $X(z=1)=1$ , where here  $X(z)$  indicates the treatment a patient would receive if (counter to fact) assigned to  $Z=z$ . Under the exclusion restriction, the null hypothesis within this sub-population that  $X$  has no effect on  $Y$  is equivalent to the null hypothesis that  $Z$  has no effect of  $Y$ . Under random assignment for the whole population, each individual in the Complier subpopulation has the same probability of being *assigned* to treatment. Thus we could use the randomization distribution of the outcomes for Compliers under the null hypothesis to carry out a significance test.

However, we face the obvious difficulty that membership in the Complier subpopulation generally cannot be determined from the observed data alone. Although we know that Compliers will have  $Z=X$ , this condition is necessary but not sufficient. For example, in the  $Z=1$  arm individuals with  $X=1$  may be either Compliers or ‘Always Takers’, where the latter subgroup are individuals who would always take the active treatment even if (counter to fact) they had been assigned to the placebo group ( $Z=0$ ). Conversely, an individual with  $Z=X=0$  may be either a Complier or someone who refuses to take treatment regardless of their assigned group, in other words a ‘Never Taker’.

If somehow we were told which individuals in the population were Compliers, then we could simply test the sharp null hypothesis by performing a significance test, such as Fisher’s exact test, on the  $(X, Y)$  sub-table, or equivalently the  $(Z, Y)$  subtable, for Compliers. One may circumvent the problem of not knowing who the Compliers are by just considering all *logically* possible values for the number of Compliers in any given  $(Z, X, Y)$  stratum that may contain them (in which  $Z=X$ ), and then carrying out the significance test for the corresponding  $(X, Y)$  subtable for the Compliers. Taking the maximum over all the resulting p-values would then give a valid p-value for the null hypothesis.

There are, however, two concerns with such an approach. The first is that such a procedure will have no or very low statistical power to reject the null hypothesis, since it is logically possible (though extremely unlikely) that there are no Compliers in a given stratum (in which  $Z=X$ ). The second is that such an approach ignores the information provided by strata that do not contain Compliers, in which  $Z \neq X$ .

We will assume that there are no patients who consistently do the opposite of their assignment, sometimes called ‘Defiers’ [3], so for all individuals:

$$X(z=0) \leq X(z=1). \quad (1)$$

It follows from this assumption that all individuals in the  $(Z=1, X=0)$  stratum are Never Takers. Under random assignment of treatment ( $Z$ ), the proportion of Never Takers in the  $Z=1$  arm should be approximately the same as in the  $Z=0$  arm. This information then reduces the range of probable values (under the randomization distribution) for the number of Compliers in the  $(Z=X=0)$  stratum.

Loh and Richardson [10], following [11], use a pre-specified significance level  $\gamma$  to construct a confidence set of values for the number of Compliers in a given  $(Z, X)$  stratum. Only values of the number of Compliers that do not indicate large imbalance between the  $Z=1$  and  $Z=0$  arms, under the randomization distribution, are used to carry out Fisher’s exact test in the implied  $(X, Y)$  table for Compliers. Taking the maximum over these p-values and adding  $\gamma$  then provides a valid but conservative p-value.

However, the procedure in [10] requires a pre-determined (non-zero) value of  $\gamma$  to eliminate ‘unlikely’ values for the number of Compliers from consideration when controlling the Type I error rate in a *hypothesis* test. The resulting p-value will hence always be greater than or equal to  $\gamma$ . This is problematic if, as in a *significance* test, we wish to interpret the p-value as measuring the strength of evidence against the null hypothesis.

In this paper we consider an alternative approach whereby we compare the ratio of the largest probability for the observed data assuming that the sharp null hypothesis holds among Compliers, with the largest probability in the case where we allow a causal effect among Compliers. Such a generalized likelihood ratio (GLR) criterion (see for example [15]) lets us evaluate whether the alternative hypothesis is a significantly better explanation for the observed dataset than the null hypothesis, even when the number of Compliers is unknown.

For a given number of Compliers, the relative frequency with which, over hypothetical replications under the null hypothesis, we would obtain a value of the GLR that is as small or smaller than that which we observed, would then be a p-value. Since this relative frequency will depend on the number of Compliers, we maximize the p-value over

the number of Compliers. This results in a valid p-value that is suitable to be used in a significance test since it can be arbitrarily close to zero (it does not require specification of some  $\gamma$ ). Furthermore, the resulting test has power against some alternatives in which there is a non-zero average causal effect among Compliers.

The remainder of the paper is organized as follows. Section 2 formalizes the potential outcome framework and sets up the motivating examples. The steps to find the maximum likelihood when the null hypothesis holds, and in general, are detailed in Section 3. Section 4 presents the generalized likelihood ratio (GLR) and describes how to find a valid frequentist p-value. The results from applying the procedure to the motivating examples are shown in Section 5. Finally, in Section 6 we briefly describe the extension to include Always Takers.

## 2 POTENTIAL OUTCOME FRAMEWORK

We now formalize the foregoing development. Recall the following:

- $Z$  is the randomized treatment assignment, where 1 indicates assignment to drug;
- $X$  is the treatment exposure subsequent to assignment, where 1 indicates drug received;
- $Y$  is the final response, where 1 indicates a desirable outcome, such as survival.

The potential outcome  $X_{z_i}$  is the treatment  $X$  a patient *would* be exposed to if assigned  $z = i$ . Using these potential outcomes we may define four generic compliance ‘types’  $t_X$  listed in Table 1. We denote the set of such types by  $\mathbb{D}_X$ .

The potential outcomes are linked to the observed outcomes by the consistency axiom [14], which requires that  $Z = z$  implies  $X = X_z$ .

Table 1: Compliance Types ( $t_X$ ) based on Potential Outcomes  $X_z$ , [8].

| $X_{z_0}$ | $X_{z_1}$ | Compliance Type $t_X$ |              |
|-----------|-----------|-----------------------|--------------|
| 0         | 0         | NT                    | Never Taker  |
| 1         | 0         | DE                    | Defier       |
| 0         | 1         | CO                    | Complier     |
| 1         | 1         | AT                    | Always Taker |

As stated above in (1) we will assume that there are no Defiers. We will also focus on the case where there are no Always Takers, so:

$$Z = 0 \Rightarrow X = 0. \tag{2}$$

This assumption will hold in studies where individuals not assigned to treatment are unable to obtain the active treat-

ment outside of the trial.

### 2.1 EXCLUSION RESTRICTION

The potential outcome for a given individual  $Y_{x_j z_i}$  is the subject’s response  $Y$  under exposure to treatment  $x = j$ , and treatment assignment  $z = i$ . Without further assumptions there are  $16 = 2^{2^2}$  possible sets of values for the variables  $(Y_{x_0 z_0}, Y_{x_1 z_0}, Y_{x_0 z_1}, Y_{x_1 z_1})$ . However, we will assume that there is no (individual-level) direct effect of  $Z$  on  $Y$  relative  $X$ , so that for  $j, i, i' \in \{0, 1\}$ , we have:

$$Y_{x_j z_i} = Y_{x_j z_{i'}} \equiv Y_{x_j}. \tag{3}$$

Assumption (3) is guaranteed to hold under double-blind placebo-controlled trials in which the active treatment is without side-effects and unavailable to patients in the control arm. The response type  $t_Y$  then simplifies to just four types, with  $\mathbb{D}_Y$  as the set of such types, shown in Table 2.

The potential outcomes for  $Y$  are again linked to the observed outcomes via the consistency axiom, so that if  $X = x$  then  $Y = Y_x$ .

Table 2: Response Types ( $t_Y$ ) under Exclusion Restriction (3), [6].

| $Y_{x_0}$ | $Y_{x_1}$ | Response Type $t_Y$ |                |
|-----------|-----------|---------------------|----------------|
| 0         | 0         | NR                  | Never Recover  |
| 1         | 0         | HU                  | Hurt           |
| 0         | 1         | HE                  | Helped         |
| 1         | 1         | AR                  | Always Recover |

### 2.2 RANDOMIZATION ASSUMPTION

We make the following assumption:

$$Z \perp\!\!\!\perp \{X_{z_0}, X_{z_1}, Y_{x_0}, Y_{x_1}\} \tag{4}$$

The assumption states that the distribution of compliance and response types ( $t_X, t_Y$ ) is the same in both the  $z = 1$  and  $z = 0$  arms; in other words, that  $Z$  is (jointly) independent of the potential outcomes. This will hold whenever treatment assignment  $Z$  is physically randomized.

### 2.3 THE INSTRUMENTAL VARIABLE (IV) MODEL

The model defined by (3) and (4) is known as the Instrumental Variable (IV) model (see for example [1]). A graph corresponding to the IV model given by (3) and (4) is shown in Figure 1. The exclusion restriction (3) corresponds to the absence of a  $Z \rightarrow Y$  edge while the randomization assumption (4) is indicated by the absence of edges directed into  $Z$ .

## 2.4 AVERAGE CAUSAL EFFECT OF X ON Y

The average causal effect (ACE) of treatment exposure  $X$  on outcome  $Y$  is defined as:

$$\text{ACE}(X \rightarrow Y) \equiv E[Y_{x_1} - Y_{x_0}]. \quad (5)$$

The ACE for the sub-population of Compliers is:

$$\text{ACE}_{CO}(X \rightarrow Y) \equiv E[Y_{x_1} - Y_{x_0} \mid t_X = CO]. \quad (6)$$

Since for Compliers  $X_z = z$ , it follows that  $Y_{X=z} \equiv Y_{X_z} = Y_z$  so that

$$\text{ACE}_{CO}(X \rightarrow Y) = \text{ITT}_{CO} \equiv E[Y_{z_1} - Y_{z_0} \mid t_X = CO], \quad (7)$$

or in words, the Average Causal Effect of  $X$  on  $Y$  for Compliers is equal to the *Intent-to-Treat effect* of  $Z$  on  $Y$  for Compliers ( $\text{ITT}_{CO}$ ).

Under the assumption (1) that there are no Defiers, the global null hypothesis  $\text{ACE}(X \rightarrow Y) = 0$  holds if and only if all the principal stratum-specific null hypotheses  $\text{ACE}_{t_X}(X \rightarrow Y) = 0$  for  $t_X \in \{NT, CO, AT\}$  jointly hold. Evidence against the (narrower) null hypothesis that  $\text{ACE}_{CO}(X \rightarrow Y) = 0$  hence implies evidence against the global null hypothesis  $\text{ACE}(X \rightarrow Y) = 0$  as well.

By definition Never Takers and Always Takers always have the same observed values of  $X = 0$  and  $X = 1$  respectively (regardless of the  $Z$  arm they are assigned to). Consequently without further experimentation (to change compliance for these individuals), there is no test for the average causal effect of  $X$  on  $Y$  in either of these principal strata. Thus assuming (1) the only sub-population for which we may observe evidence that  $\text{ACE}_{t_X}(X \rightarrow Y) \neq 0$  are the Compliers ( $CO$ ).<sup>1</sup>

Furthermore, with the added assumption that there are no Always Takers, the ‘treated’ sub-population are simply the Compliers, such that the test of  $\text{ACE}_{CO}(X \rightarrow Y) = 0$  is the same test for the effect of treatment on the treated,  $E[Y_{x_1} - Y_{x_0} \mid X = 1] = 0$ .

## 2.5 MOTIVATING EXAMPLES

We consider two examples of randomized experiments with noncompliance. The first dataset is from a psychology experiment where individuals were randomly assigned to one of two groups (Table 3). The treatment group was offered a small cup of pop soda ( $Z = 1$ ), while the placebo group was offered a small cup of water ( $Z = 0$ ). Compliance was whether the individual consumed the offered soda ( $X = 1$ ) or not ( $X = 0$ ). Individuals who were not offered soda in the control group ( $Z = 0$ ) had no access to soda, as this was a closed study. There are thus two structural zeros, since

<sup>1</sup>This is why, even though our procedure is a test of the global null, we describe it as a test of the sharp null for Compliers.

$Z = 0$  implies  $X = 0$ . The response was a binary variable of whether the subject disposed of the cup after the session ( $Y = 1$ ) or left the cup on the table ( $Y = 0$ ). If we were to test the null hypothesis of  $Z \perp\!\!\!\perp Y$  with Fisher’s Exact Test for the corresponding  $2 \times 2$  table, we would get a p-value of 0.0085. However, if we disregarded the 30 individuals in the  $(Z = 1, X = 0)$  stratum and just tested  $Z \perp\!\!\!\perp Y$  among the  $(Z = X)$  stratum, we would get a p-value of 0.0546. Finally, Fisher’s Exact Test for the null hypothesis that  $X \perp\!\!\!\perp Y$  gives a p-value of 0.2157.

Table 3: Psychology Data

| $z$ | $x$ | $y$ | count | $z$ | $x$ | $y$ | count |
|-----|-----|-----|-------|-----|-----|-----|-------|
| 0   | 0   | 0   | 53    | 1   | 0   | 0   | 13    |
| 0   | 0   | 1   | 23    | 1   | 0   | 1   | 17    |
| 0   | 1   | 0   | 0     | 1   | 1   | 0   | 24    |
| 0   | 1   | 1   | 0     | 1   | 1   | 1   | 23    |

The second dataset is from a double-blind placebo-controlled randomized trial of Cholestyramine [4]. Subjects were randomly assigned to one of two arms: subjects in the treatment arm were prescribed Cholestyramine ( $Z = 1$ ), and those in the other arm were given a placebo ( $Z = 0$ ). Compliance was a continuous measure tracking the quantity of prescribed dosage consumed, over several years of treatment during the trial. The response was the average post-treatment cholesterol level, and also a continuous variable. Both continuous measures were dichotomized in [13], and the resulting counts are shown in Table 4. There are also two structural zeros in this dataset, since subjects who are not assigned treatment in the control arm ( $Z = 0$ ) could not obtain the experimental drug Cholestyramine.

Table 4: Cholestyramine/Lipid Data

| $z$ | $x$ | $y$ | count | $z$ | $x$ | $y$ | count |
|-----|-----|-----|-------|-----|-----|-----|-------|
| 0   | 0   | 0   | 158   | 1   | 0   | 0   | 52    |
| 0   | 0   | 1   | 14    | 1   | 0   | 1   | 12    |
| 0   | 1   | 0   | 0     | 1   | 1   | 0   | 23    |
| 0   | 1   | 1   | 0     | 1   | 1   | 1   | 78    |

For both studies in terms of the compliance types, there are no Defiers and no Always Takers, and both (1) and (2) hold. Furthermore, since both studies were double-blind randomized control trials, it may be safely assumed that  $Z$  has no effect on  $Y$  other than through  $X$ , so that the exclusion restriction (3) holds. Thus in this case, there are four response types  $t_Y$ , but only two compliance types  $t_X$ , which gives us eight combinations for  $(t_X, t_Y) \in \{NT, CO\} \times \{HE, HU, AR, NR\}$ . We will consider this simpler case during our main development, though the approach extends to the more general case in which there are also Always Takers.



### 3 MAXIMIZING THE LIKELIHOOD UNDER RANDOMIZATION

We first introduce the notation. Let  $n_{y_k x_j z_i}$  be the observable count of the number of individuals in the finite population who are assigned to treatment  $z = i$ , with exposure  $x = j$  and outcome  $y = k$ . We denote marginal tables similarly, for example  $n_{y_k}$  and  $n_{z_i}$ .

Let  $n_{t_Y, z_i}^{t_X}$  be the number of individuals in the finite population of compliance type  $t_X$  and response type  $t_Y$ , who are assigned to treatment  $z = i$ . Similarly, let  $n_{y_k z_i}^{t_X}$  be the number of individuals of compliance type  $t_X$  who are observed to have outcome  $y = k$  in the  $z = i$  arm, and  $n_{y_k}^{t_X} \equiv n_{y_k z_0}^{t_X} + n_{y_k z_1}^{t_X}$  be the total number of individuals in the finite population of compliance type  $t_X$  with observed outcome  $y = k$ . It should be noted that the counts  $n_{t_Y, z_i}^{t_X}$ ,  $n_{y_k z_i}^{t_X}$  and  $n_{y_k}^{t_X}$  are not all point-identified since they may not be directly observable from the data.

Our interest lies in testing the individual level (or ‘sharp’) causal null hypothesis that there is no effect of  $X$  on  $Y$  amongst Compliers:

$$H_0 : Y_{x_0} = Y_{x_1}. \quad (8)$$

Under the sharp null hypothesis (8), within the Complier sub-population, each individual would have the same observed outcome  $Y$  regardless of whether they took the treatment ( $X = Z = 1$ ) or did not do so ( $X = Z = 0$ ). Note that if the individual level causal null hypothesis (8) holds, then there is a zero average causal effect of  $X$  on  $Y$  for the sub-population of Compliers (CO) and  $ACE_{CO}(X \rightarrow Y) = 0$ .

Thus under the null (8), the number of Compliers with observed responses  $y = 0$  and  $y = 1$  are just the number of Compliers of types Never Recover (NR) and Always Recover (AR) respectively:

$$\begin{aligned} n_{y_0}^{CO} & \underset{H_0}{=} n_{NR, z_0}^{CO} + n_{NR, z_1}^{CO}, \\ n_{y_1}^{CO} & \underset{H_0}{=} n_{AR, z_0}^{CO} + n_{AR, z_1}^{CO}. \end{aligned}$$

If the number of Compliers assigned to  $z = 1$  vs.  $z = 0$  were pre-specified in advance by the experimental design then, over hypothetical replications, the margins of the  $2 \times 2$  sub-table for Compliers containing the four counts  $n_{t_Y, z_i}^{CO}$  for  $t_Y \in \{NR, AR\}$ ,  $i \in \{0, 1\}$  would be fixed. The resulting distribution for one of the cells, for example  $n_{AR, z_1}^{CO}$ , would be a hypergeometric distribution under the null hypothesis.

However, since we have no way to ensure a specific number of Compliers are assigned to treatment (or control) this may vary over hypothetical replications, hence none of the four counts  $n_{t_Y, z_i}^{CO}$  in the subtable for Compliers will follow a hypergeometric distribution. Further these counts are not directly observable from the data.

### 3.1 NUISANCE PARAMETERS

Denote  $\psi_k^{NT}$  as the total number of Never Takers with observed outcome  $y = k$ , such that the bivariate parameter  $\psi$  is:

$$\psi \equiv (\psi_0^{NT} \equiv n_{y_0}^{NT}, \psi_1^{NT} \equiv n_{y_1}^{NT}).$$

Figure 2 describes the sum relationships between the observed dataset  $\{n_{y_k x_j z_i}\}$  and counts  $n_{t_Y, z_i}^{t_X}$ ,  $n_{y_k z_i}^{t_X}$  and  $n_{y_k}^{t_X}$  under the null (8).

The counts  $n_{NR, z_1}^{CO}$  and  $n_{AR, z_1}^{CO}$  in the treatment arm ( $z = 1$ ) are directly observable from the data as  $n_{y_0 x_1 z_1}$  and  $n_{y_1 x_1 z_1}$  respectively. However, the presence of Never Takers in the finite population prevents us from point-identifying  $n_{NR, z_0}^{CO}$  and  $n_{AR, z_0}^{CO}$  in the placebo arm ( $z = 0$ ).

The unknown number of Never Takers  $\psi \equiv (\psi_0^{NT}, \psi_1^{NT})$  may thus be regarded as ‘nuisance parameters’, since if we knew these quantities, we could simply determine the unobservable counts for the Never Takers in the  $z_0$  arm:

$$\begin{aligned} n_{y_0, z_0}^{NT} & \equiv \psi_0^{NT} - n_{y_0, z_1}^{NT} = \psi_0^{NT} - n_{y_0 x_0 z_1}, \\ n_{y_1, z_0}^{NT} & \equiv \psi_1^{NT} - n_{y_1, z_1}^{NT} = \psi_1^{NT} - n_{y_1 x_0 z_1}. \end{aligned}$$

This in turn tells us what the exact values of  $n_{NR, z_0}^{CO}$  and  $n_{AR, z_0}^{CO}$  are, since  $n_{NR, z_0}^{CO}$  and  $n_{y_0, z_0}^{NT}$  add up to the observable quantity  $n_{y_0 x_0 z_0}$ , and similarly,  $n_{AR, z_0}^{CO}$  and  $n_{y_1, z_0}^{NT}$  add up to  $n_{y_1 x_0 z_0}$ .

Since  $\psi_0^{NT}$  and  $\psi_1^{NT}$  are bounded by the observable quantities in the data  $\{n_{y_k x_j z_i}\}$ , the space of possible values for the nuisance parameter  $\psi$  is the Cartesian product of the respective one-dimensional ranges for  $\psi_0$  and  $\psi_1$ :

$$\begin{aligned} \psi_0^{NT} & \in [n_{y_0 x_0 z_1}, n_{y_0 x_0 z_1} + n_{y_0 x_0 z_0}] = \Psi_0, \\ \psi_1^{NT} & \in [n_{y_1 x_0 z_1}, n_{y_1 x_0 z_1} + n_{y_1 x_0 z_0}] = \Psi_1, \\ \Psi & = \Psi_0 \times \Psi_1. \end{aligned} \quad (9)$$

### 3.2 MAXIMIZING THE HYPERGEOMETRIC PROBABILITY IN A $2 \times 2$ TABLE

Before analyzing the likelihood in our specific problem, we review the following related result: Suppose an urn contains  $N$  balls that are either red or green.  $N - k$  balls are drawn from the urn, of which  $b$  balls are red. What is the most likely number of red balls  $x$  remaining in the urn, or equivalently, the most likely total number of red balls  $b + x$  in the urn initially?

Table 5:  $2 \times 2$  Table With Unknown Column Totals

|           | Red     | Green         | Row     |
|-----------|---------|---------------|---------|
| Not drawn | $x$     | $k - x$       | $k$     |
| Drawn     | $b$     | $(N - k) - b$ | $N - k$ |
| Column    | $b + x$ | $N - (b + x)$ | $N$     |

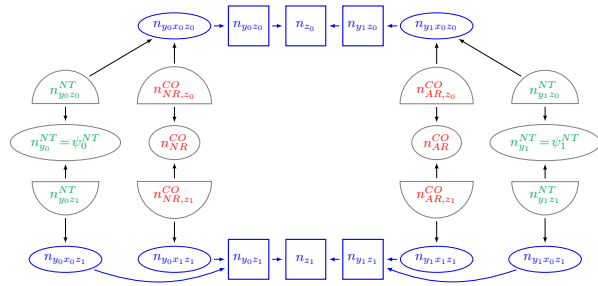


Figure 2: Sum Relationships between the Observed Dataset  $\{n_{y_k, x_j, z_i}\}$  and Possibly Unobserved Counts  $n_{t_Y, z_i}^{tX}$ ,  $n_{y_k, z_i}^{tX}$ ,  $n_{y_k}^{tX}$ , Assuming  $H_0$  (8); See Table 6.

We would thus like to maximize the hypergeometric probability corresponding to Table 5 with respect to  $x$ :

$$\Pr(x | (k, b, N)) = \binom{b+x}{x} \binom{N-(b+x)}{k-x} / \binom{N}{k}.$$

When  $b = 0$ , the most likely value of  $x$  would just be 0 as well. So if all  $N - k$  balls drawn were green, then the most likely number of red balls in the urn is 0.

**Theorem 1.** *In a  $2 \times 2$  table where the row totals  $(k, N - k)$  and the counts in one row  $(b, (N - k) - b)$  are fixed, the most likely value of  $x \in [0, k]$  under the randomization assumption is:*

$$\hat{x} = \arg \max_{x \in [0, k]} \left\{ x < (k+1) \frac{b}{N-k} \right\} = \left\lfloor \left\lceil (k+1) \frac{b}{N-k} \right\rceil \right\rfloor,$$

where the ‘basement’ function  $\lfloor \lceil a \rceil \rfloor$  is defined as:

$$\lfloor \lceil a \rceil \rfloor = \max\{0, \lceil a \rceil - 1\}.$$

Equivalently,

$$b + \hat{x} = \begin{cases} \left\lfloor b \frac{N}{N-k} \right\rfloor & \text{if } b \frac{N+1}{N-k} \leq \left\lceil b \frac{N}{N-k} \right\rceil, \\ \left\lceil b \frac{N}{N-k} \right\rceil & \text{otherwise.} \end{cases}$$

The proof for Theorem 1 is given in the supplementary material.

### 3.3 MAXIMUM LIKELIHOOD UNDER THE NULL

For some given value of the nuisance parameters  $\psi = \mathbf{u}$ , the counts  $n_{t_Y, z_i}^{tX}$ ,  $n_{y_k, z_i}^{tX}$  and  $n_{y_k}^{tX}$  are all point-identified from the observed dataset  $\{n_{y_k, x_j, z_i}\}$ . We may hence describe the exact counts in a  $2 \times 4$  full contingency table such as Table 6.

When we fix the value of  $\psi$  at the value  $\mathbf{u}$ , the total number of Never Takers with observed outcomes  $y = 0$  and  $y = 1$  ( $u_0^{NT}$  and  $u_1^{NT}$  respectively), as well as the total number

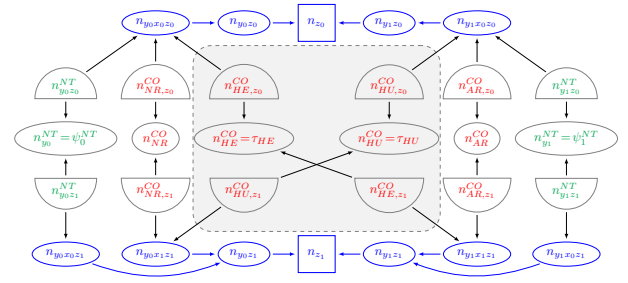


Figure 3: Sum Relationships between the Observed Dataset  $\{n_{y_k, x_j, z_i}\}$  and Possibly Unobserved Counts  $n_{t_Y, z_i}^{tX}$ ,  $n_{y_k, z_i}^{tX}$ ,  $n_{y_k}^{tX}$ , Without Assuming  $H_0$  (8); See Table 7.

of Compliers with observed responses  $y = 0$  and  $y = 1$  ( $n_{y_0} - u_0^{NT}$  and  $n_{y_1} - u_1^{NT}$  respectively) are fixed in the population, and would not change under  $H_0$  as we vary over all possible assignments of individuals to  $z = 0$  and  $z = 1$ .

Given the fixed column and row totals in Table 6 over repeated samplings, the randomization distribution under the null hypothesis (8) for the subjects assigned to the  $z = 1$  arm is thus the multiple hypergeometric distribution [9, Chapter 39]:

$$\begin{aligned} \Pr(\{n_{y_k, x_j, z_i}\} | \psi = \mathbf{u}, H_0) \\ = \frac{\binom{u_0^{NT}}{n_{y_0, x_0, z_1}} \binom{u_1^{NT}}{n_{y_1, x_0, z_1}} \binom{n_{y_0} - u_0^{NT}}{n_{y_0, x_1, z_1}} \binom{n_{y_1} - u_1^{NT}}{n_{y_1, x_1, z_1}}}{\binom{N}{n_{z_0}}} \end{aligned} \quad (10)$$

Note that the sharp null hypothesis for Compliers (8) holding places no restriction on the range of values for the nuisance parameter  $\psi$ . We shall thus consider the value of the nuisance parameter that lends the strongest support under  $H_0$  to the observed dataset  $\{n_{y_k, x_j, z_i}\}$ , by finding the maximum likelihood with respect to  $\psi$ :

$$q^{H_0}(\{n_{y_k, x_j, z_i}\}) = \max_{\psi \in \Psi} \Pr(\{n_{y_k, x_j, z_i}\} | \psi, H_0). \quad (11)$$

An exhaustive search over the two-dimensional discrete grid of the parameter space  $\Psi$  would require calculating  $|\Psi| = (n_{y_0, x_0, z_0} + 1) \times (n_{y_1, x_0, z_0} + 1)$  different hypergeometric probabilities.<sup>2</sup>

Instead, we partition Table 6 into two variation-independent  $2 \times 2$  subtables: one for the Never Takers and Compliers with observed  $y = 0$  outcomes (types  $(NT, y_0)$  and  $(CO, NR)$  respectively), and another for the Compliers and Never Takers with observed  $y = 1$  outcomes (types  $(CO, AR)$  and  $(NT, y_1)$  respectively). The joint probability (10) then factorizes into the corresponding functions of  $\psi_0^{NT}$  and  $\psi_1^{NT}$  below:

<sup>2</sup>For example in the Lipid data, the search space would be of size  $(158 + 1) \times (14 + 1) = 2,385$ .

$$\Pr(\{n_{y_k x_j z_i}\} | \boldsymbol{\psi}, H_0) = \frac{\binom{n_{y_0}}{n_{y_0 x_0 z_0}} \binom{n_{y_1}}{n_{y_1 x_0 z_0}}}{\binom{N}{n_{z_0}}} \times g_0(\psi_0^{NT} | \{n_{y_k x_j z_i}\}) \times g_1(\psi_1^{NT} | \{n_{y_k x_j z_i}\}); \quad (12)$$

$$g_0(\psi_0^{NT} | \{n_{y_k x_j z_i}\}) = \frac{\binom{\psi_0^{NT}}{n_{y_0 x_0 z_1}} \binom{n_{y_0} - \psi_0^{NT}}{n_{y_0 x_1 z_1}}}{\binom{n_{y_0}}{n_{y_0 x_0 z_0}}} \quad (13)$$

$$g_1(\psi_1^{NT} | \{n_{y_k x_j z_i}\}) = \frac{\binom{\psi_1^{NT}}{n_{y_1 x_0 z_1}} \binom{n_{y_1} - \psi_1^{NT}}{n_{y_1 x_1 z_1}}}{\binom{n_{y_1}}{n_{y_1 x_0 z_0}}}. \quad (14)$$

In both subtables, the cell counts in the  $z = 1$  arm are fixed, while the row totals for the  $z = 0$  arm are  $n_{y_0 x_0 z_0}$  and  $n_{y_1 x_0 z_0}$  respectively. We may then apply Theorem 1 directly to each subtable to find the following values of  $\psi_0^{NT}$  and  $\psi_1^{NT}$  that maximise the respective hypergeometric probabilities (13) and (14).

$$\widehat{\psi}_0^{NT} = \begin{cases} \left\lfloor \frac{n_{y_0} n_{y_0 x_0 z_1}}{n_{y_0} - n_{y_0 x_0 z_0}} \right\rfloor & \text{if } \frac{(n_{y_0} + 1) n_{y_0 x_0 z_1}}{n_{y_0} - n_{y_0 x_0 z_0}} \leq \left\lceil \frac{n_{y_0} n_{y_0 x_0 z_1}}{n_{y_0} - n_{y_0 x_0 z_0}} \right\rceil, \\ \left\lfloor \frac{n_{y_0} n_{y_0 x_0 z_1}}{n_{y_0} - n_{y_0 x_0 z_0}} \right\rfloor & \text{otherwise;} \end{cases}$$

$$\widehat{\psi}_1^{NT} = \begin{cases} \left\lfloor \frac{n_{y_1} n_{y_1 x_0 z_1}}{n_{y_1} - n_{y_1 x_0 z_0}} \right\rfloor & \text{if } \frac{(n_{y_1} + 1) n_{y_1 x_0 z_1}}{n_{y_1} - n_{y_1 x_0 z_0}} \leq \left\lceil \frac{n_{y_1} n_{y_1 x_0 z_1}}{n_{y_1} - n_{y_1 x_0 z_0}} \right\rceil, \\ \left\lfloor \frac{n_{y_1} n_{y_1 x_0 z_1}}{n_{y_1} - n_{y_1 x_0 z_0}} \right\rfloor & \text{otherwise.} \end{cases}$$

The largest value of the probability of the observed dataset under the null (11) is then:

$$q^{H_0}(\{n_{y_k x_j z_i}\}) = \Pr(\{n_{y_k x_j z_i}\} | (\widehat{\psi}_0^{NT}, \widehat{\psi}_1^{NT}), H_0).$$

### 3.4 MAXIMUM LIKELIHOOD UNDER THE ALTERNATIVE

When the null hypothesis does not hold, there may be individuals in the Complier sub-population whose treatment exposure  $X = j$  has an effect on their observed outcome  $Y = k$ . Compliers with observed responses  $y = 0$  are no longer limited to being only of response type Never Recover ( $NR$ ): they may also be of types Helped (in the  $z = 0$  arm) or Hurt (in the  $z = 1$  arm). Similarly, Compliers with observed responses  $y = 1$  may also be one of three response types: Always Recover ( $AR$ ), Helped (in the  $z = 1$  arm) or Hurt (in the  $z = 0$  arm).

Denote by  $\tau_i^{t_Y}(\boldsymbol{\psi})$  the number of Compliers in the finite population with response type  $t_Y$  assigned to treatment  $z = i$ , for some fixed value of  $\boldsymbol{\psi}$ . For example,  $\tau_0^{HE}(\boldsymbol{\psi})$  is the number of Compliers of type Helped in the  $z = 0$  arm. The parameter vector  $\boldsymbol{\tau}(\boldsymbol{\psi})$  is then:

$$\boldsymbol{\tau}(\boldsymbol{\psi}) \equiv \left( \tau_0^{HE}(\boldsymbol{\psi}) \equiv n_{HE,z_0}^{CO}(\boldsymbol{\psi}), \quad \tau_0^{HU}(\boldsymbol{\psi}) \equiv n_{HU,z_0}^{CO}(\boldsymbol{\psi}), \right. \\ \left. \tau_1^{HE} \equiv n_{HE,z_1}^{CO}, \quad \tau_1^{HU} \equiv n_{HU,z_1}^{CO} \right).$$

The sum relationships between the observed dataset  $\{n_{y_k x_j z_i}\}$  and counts  $n_{t_Y, z_i}^{t_X}$ ,  $n_{y_k, z_i}^{t_X}$  and  $n_{y_k}^{t_X}$  may then be described in Figure 3.

The space of possible values for the parameter  $\boldsymbol{\tau}(\boldsymbol{\psi})$  depends on the fixed value of  $\boldsymbol{\psi}$  and corresponds to a four-dimensional discrete grid:

$$\begin{aligned} \tau_0^{HE}(\boldsymbol{\psi}) &\in [0, n_{y_0 x_0 z_0} - (\psi_0^{NT} - n_{y_0 x_0 z_1})] \equiv \mathbf{T}_0^{HE}(\boldsymbol{\psi}), \\ \tau_0^{HU}(\boldsymbol{\psi}) &\in [0, n_{y_1 x_0 z_0} - (\psi_1^{NT} - n_{y_1 x_0 z_1})] \equiv \mathbf{T}_0^{HU}(\boldsymbol{\psi}), \\ \tau_1^{HE} &\in [0, n_{y_1 x_1 z_1}] \equiv \mathbf{T}_1^{HE}, \\ \tau_1^{HU} &\in [0, n_{y_0 x_1 z_1}] \equiv \mathbf{T}_1^{HU}, \\ \mathbf{T}(\boldsymbol{\psi}) &= \mathbf{T}_0^{HE}(\boldsymbol{\psi}) \times \mathbf{T}_0^{HU}(\boldsymbol{\psi}) \times \mathbf{T}_1^{HE} \times \mathbf{T}_1^{HU}. \end{aligned} \quad (15)$$

For some given value of the nuisance parameters  $\boldsymbol{\psi} = \mathbf{u} \equiv (u_0^{NT}, u_1^{NT})$ , and the primary parameters  $\boldsymbol{\tau}(\mathbf{u}) = \mathbf{t}(\mathbf{u}) \equiv (t_0^{HE}(\mathbf{u}), t_0^{HU}(\mathbf{u}), t_1^{HE}, t_1^{HU})$ , the counts  $n_{t_Y, z_i}^{t_X}$ ,  $n_{y_k, z_i}^{t_X}$  and  $n_{y_k}^{t_X}$  are all point-identified from the observed dataset  $\{n_{y_k x_j z_i}\}$ . The exact counts may be summarized in a  $2 \times 6$  full contingency table such as Table 7.

Given the fixed column and row totals in Table 7 over repeated samplings, the multiple hypergeometric probability of the subjects assigned to the  $z = 1$  arm, when we no longer assume  $H_0$  to hold, is:

$$\begin{aligned} \Pr(\{n_{y_k x_j z_i}\} | (\boldsymbol{\psi}, \boldsymbol{\tau}(\boldsymbol{\psi})) = (\mathbf{u}, \mathbf{t}(\mathbf{u}))) &= \frac{\binom{u_0^{NT}}{n_{y_0 x_0 z_1}} \binom{t_0^{HE}(\mathbf{u}) + t_1^{HE}}{t_1^{HE}} \binom{t_0^{HU}(\mathbf{u}) + t_1^{HU}}{t_1^{HU}} \binom{u_1^{NT}}{n_{y_1 x_0 z_1}}}{\binom{N}{n_{z_0}}} \\ &\times \binom{n_{y_0} - t_0^{HE}(\mathbf{u}) - t_1^{HU} - u_0^{NT}}{n_{y_0 x_1 z_1} - t_1^{HU}} \binom{n_{y_1} - t_0^{HU}(\mathbf{u}) - t_1^{HE} - u_1^{NT}}{n_{y_1 x_1 z_1} - t_1^{HE}}. \end{aligned} \quad (16)$$

The maximum likelihood for the observed data  $\{n_{y_k x_j z_i}\}$ , allowing for Compliers who are Helped and Hurt, is then:

$$\begin{aligned} q^{ML}(\{n_{y_k x_j z_i}\}) &= \max_{\boldsymbol{\psi} \in \boldsymbol{\Psi}} \max_{\boldsymbol{\tau}(\boldsymbol{\psi}) \in \mathbf{T}(\boldsymbol{\psi})} \Pr(\{n_{y_k x_j z_i}\} | (\boldsymbol{\psi}, \boldsymbol{\tau}(\boldsymbol{\psi}))). \end{aligned}$$

Similar to the maximization procedure under the null, we would like to circumvent an exhaustive search of the parameter space  $\{\boldsymbol{\Psi} \times \mathbf{T}(\boldsymbol{\Psi})\}$  by decomposing the joint probability (16) into separate objective functions. However, unlike the full contingency table under  $H_0$  in Table 6, we cannot simply partition Table 7 into variation-independent subtables based only on the observed  $y = 0$  and  $y = 1$  outcomes. This is because if there was an effect of the treatment  $X$  on  $Y$ , then there is a Complier individual of type Helped or Hurt who would have had a different outcome  $Y$  had they been assigned to a different level of  $Z$ , and hence received a different exposure level  $X$ .

However, when we fix the number of Compliers of types Helped and Hurt in the  $z = 1$  arm at some value  $(\tau_1^{HE}, \tau_1^{HU}) = (t_1^{HE}, t_1^{HU})$ , all six counts in the  $z = 1$  arm are now point-identified and fixed. Then Table 7 may be partitioned into two variation-independent

Table 6: Full Contingency Table Under  $H_0$  with Cell Counts that are Point-Identified Given a Value of  $\psi = \mathbf{u}$ .

|        | $NT, y_0 \equiv$<br>$NT, (NR/HE)$ | $CO, NR$   | $CO, AR$   | $NT, y_1 \equiv$<br>$NT, (AR/HU)$ | Row       |
|--------|-----------------------------------|--|--|-----------------------------------|-----------|
| $z_0$  | $u_0^{NT} - n_{y_0 x_0 z_1}$      | $n_{y_0 x_0 z_0} - [u_0^{NT} - n_{y_0 x_0 z_1}]$ | $n_{y_1 x_0 z_0} - [u_1^{NT} - n_{y_1 x_0 z_1}]$ | $u_1^{NT} - n_{y_1 x_0 z_1}$      | $n_{z_0}$ |
| $z_1$  | $n_{y_0 x_0 z_1}$                 | $n_{y_0 x_1 z_1}$                                | $n_{y_1 x_1 z_1}$                                | $n_{y_1 x_0 z_1}$                 | $n_{z_1}$ |
| Column | $u_0^{NT}$                        | $n_{y_0} - u_0^{NT}$                             | $n_{y_1} - u_1^{NT}$                             | $u_1^{NT}$                        | $N$       |

Table 7: Full Contingency Table Allowing for Helped and Hurt with Cell Counts that are Point-Identified Given Values of  $\psi = \mathbf{u}$  and  $\boldsymbol{\tau}(\mathbf{u}) = \mathbf{t}(\mathbf{u}) \equiv (t_0^{HE}(\mathbf{u}), t_0^{HU}(\mathbf{u}), t_1^{HE}, t_1^{HU})$ .

|       | $NT, y_0 \equiv$<br>$NT, (NR/HE)$ | $CO, NR$  | $CO, HE$                          | $CO, HU$                          | $CO, AR$  | $NT, y_1 \equiv$<br>$NT, (AR/HU)$ | Row       |
|-------|-----------------------------------|---|-----------------------------------|-----------------------------------|---|-----------------------------------|-----------|
| $z_0$ | $u_0^{NT} - n_{y_0 x_0 z_1}$      | $n_{y_0 x_0 z_0} - t_0^{HE}(\mathbf{u}) - [u_0^{NT} - n_{y_0 x_0 z_1}]$ | $t_0^{HE}(\mathbf{u})$            | $t_0^{HU}(\mathbf{u})$            | $n_{y_1 x_0 z_0} - t_0^{HU}(\mathbf{u}) - [u_1^{NT} - n_{y_1 x_0 z_1}]$ | $u_1^{NT} - n_{y_1 x_0 z_1}$      | $n_{z_0}$ |
| $z_1$ | $n_{y_0 x_0 z_1}$                 | $n_{y_0 x_1 z_1} - t_1^{HU}$  | $t_1^{HE}$                        | $t_1^{HU}$                        | $n_{y_1 x_1 z_1} - t_1^{HE}$  | $n_{y_1 x_0 z_1}$                 | $n_{z_1}$ |
|       | $u_0^{NT}$                        | $n_{y_0} - t_0^{HE}(\mathbf{u}) - t_1^{HU} - u_0^{NT}$                  | $t_0^{HE}(\mathbf{u}) + t_1^{HE}$ | $t_0^{HU}(\mathbf{u}) + t_1^{HU}$ | $n_{y_1} - t_0^{HU}(\mathbf{u}) - t_1^{HE} - u_1^{NT}$                  | $u_1^{NT}$                        | $N$       |

$2 \times 3$  subtables: one for individuals of types  $(NT, y_0)$ ,  $(CO, NR)$  and  $(CO, HE)$ , and another for individuals of types  $(CO, HU)$ ,  $(CO, AR)$  and  $(NT, y_1)$ .

Given a fixed value of  $(t_1^{HE}, t_1^{HU})$ , the joint probability (16) then decomposes into a product of functions of  $(\psi_0^{NT}, \tau_0^{HE})$  and  $(\psi_1^{NT}, \tau_0^{HU})$ ; see (18) and (19) below. For the given value of  $(t_1^{HE}, t_1^{HU})$ , the cell counts in the  $z=1$  arm are fixed in each  $2 \times 3$  variation-independent subtable, while the row totals for the  $z=0$  arms are  $n_{y_0 x_0 z_0}$  and  $n_{y_1 x_0 z_0}$  respectively.

$$\begin{aligned} & \Pr(\{n_{y_k x_j z_i}\} | \psi, \tau_0^{HE}(\psi), \tau_0^{HU}(\psi), t_1^{HE}, t_1^{HU}) \\ &= \frac{\binom{n_{y_0} + t_1^{HE} - t_1^{HU}}{n_{y_0 x_0 z_0}} \binom{n_{y_1} + t_1^{HU} - t_1^{HE}}{n_{y_1 x_0 z_0}}}{\binom{N}{n_{z_0}}} \\ & \quad \times h_0(\psi_0^{NT}, \tau_0^{HE}(\psi_0^{NT}) | t_1^{HE}, t_1^{HU}, \{n_{y_k x_j z_i}\}) \\ & \quad \times h_1(\psi_1^{NT}, \tau_0^{HU}(\psi_1^{NT}) | t_1^{HE}, t_1^{HU}, \{n_{y_k x_j z_i}\}); \end{aligned} \quad (17)$$

$$\begin{aligned} & h_0(\psi_0^{NT}, \tau_0^{HE}(\psi_0^{NT}) | t_1^{HE}, t_1^{HU}, \{n_{y_k x_j z_i}\}) \\ &= \frac{\binom{\psi_0^{NT}}{n_{y_0 x_0 z_1}} \binom{\tau_0^{HE}(\psi_0^{NT}) + t_1^{HE}}{t_1^{HE}} \binom{n_{y_0} - \tau_0^{HE}(\psi_0^{NT}) - t_1^{HU} - \psi_0^{NT}}{n_{y_0 x_0 z_0}}}{\binom{n_{y_0} + t_1^{HE} - t_1^{HU}}{n_{y_0 x_0 z_0}}}, \end{aligned} \quad (18)$$

$$\begin{aligned} & h_1(\psi_1^{NT}, \tau_0^{HU}(\psi_1^{NT}) | t_1^{HE}, t_1^{HU}, \{n_{y_k x_j z_i}\}) \\ &= \frac{\binom{\psi_1^{NT}}{n_{y_1 x_0 z_1}} \binom{\tau_0^{HU}(\psi_1^{NT}) + t_1^{HU}}{t_1^{HU}} \binom{n_{y_1} - \tau_0^{HU}(\psi_1^{NT}) - t_1^{HE} - \psi_1^{NT}}{n_{y_1 x_1 z_1} - t_1^{HE}}}{\binom{n_{y_1} + t_1^{HU} - t_1^{HE}}{n_{y_1 x_0 z_0}}}. \end{aligned} \quad (19)$$

Since the  $2 \times 3$  subtables are now variation-independent, we

may find the values of:

$$\begin{aligned} & (\hat{\psi}_0^{NT}(t_1^{HE}, t_1^{HU}), \hat{\tau}_0^{HE}(\hat{\psi}_0^{NT}; t_1^{HE}, t_1^{HU})), \\ & (\hat{\psi}_1^{NT}(t_1^{HE}, t_1^{HU}), \hat{\tau}_0^{HU}(\hat{\psi}_1^{NT}; t_1^{HE}, t_1^{HU})) \end{aligned}$$

that maximise the respective conditional hypergeometric probabilities (18) and (19).

For each fixed value of  $(t_1^{HE}, t_1^{HU})$ , a naïve search over the discrete parameter space in each induced  $2 \times 3$  subtable would involve maximizing over  $\binom{n_{y_0 x_0 z_0} + 2}{2}$  and  $\binom{n_{y_1 x_0 z_0} + 2}{2}$  hypergeometric probabilities respectively.<sup>3</sup>

Instead, we apply the result from [12], which provides an algorithm to find the most likely values of the cells in the  $z_0$  arms of both subtables, *without calculating any hypergeometric probabilities*. Finally, we need only maximize over the parameter spaces for  $\tau_1^{HE}$  and  $\tau_1^{HU}$ , where there are  $|\mathbf{T}_1^{HE}| \times |\mathbf{T}_1^{HU}| = (n_{y_1 x_1 z_1} + 1) \times (n_{y_0 x_1 z_1} + 1)$  possible combinations for the point-identified counts in the  $Z=1$  arm.<sup>4</sup>

Allowing Compilers who are Helped and Hurt, the maximum likelihood for the observed dataset  $\{n_{y_k x_j z_i}\}$  is then:

$$\begin{aligned} & q^{ML}(\{n_{y_k x_j z_i}\}) = \\ & \max_{\substack{(\tau_1^{HE}, \tau_1^{HU}) \\ \in \mathbf{T}_1^{HE} \times \mathbf{T}_1^{HU}}} \Pr(\{n_{y_k x_j z_i}\} | \hat{\psi}, \tau_0^{HE}(\hat{\psi}), \\ & \quad \tau_0^{HU}(\hat{\psi}), \tau_1^{HE}, \tau_1^{HU}). \end{aligned} \quad (20)$$

<sup>3</sup>For example in the Lipid data, the search spaces would be of sizes  $\binom{158+2}{2} = 12,720$  and  $\binom{14+2}{2} = 120$  respectively.

<sup>4</sup>In the Lipid data example, we would need to calculate only  $(78 + 1) \times (23 + 1) = 1896$  hypergeometric probabilities.

## 4 GLR AND P-VALUE

A generalized likelihood ratio (GLR) lets us assess the evidence in the observed data both for *and against* the null hypothesis (8) respectively, by comparing the best possible fit of the observed data when  $H_0$  holds, against the best fit without the constraint of  $H_0$ . The generalized likelihood ratio for the observed dataset  $\{n_{y_k x_j z_i}\}$  is defined as:

$$G(\{n_{y_k x_j z_i}\}) = \frac{q^{H_0}(\{n_{y_k x_j z_i}\})}{q^{ML}(\{n_{y_k x_j z_i}\})}. \quad (21)$$

However, the distribution of the test statistic (21) under  $H_0$  depends on the chosen values of the column totals in the full contingency table (Table 6), which in turn correspond to some value of the nuisance parameter  $\psi$ . Under  $H_0$  the value of  $\psi = \mathbf{u}$  is sufficient to determine the distribution of (21) since the margin totals in the full contingency table (Table 6) are now fixed.

We may then enumerate all possible assignments, and consolidate each assignment to obtain the associated *possibly observable dataset*  $\{\tilde{n}_{y_k x_j z_i}\}$  based on the sum relationships depicted in Figure 2:

$$\begin{aligned} \tilde{n}_{y_0 x_0 z_0} &= \tilde{n}_{y_0, z_0}^{NT} + \tilde{n}_{NR, z_0}^{CO}; \tilde{n}_{y_1 x_0 z_0} = \tilde{n}_{y_1, z_0}^{NT} + \tilde{n}_{AR, z_0}^{CO}; \\ \tilde{n}_{y_0 x_0 z_1} &= u_0^{NT} - \tilde{n}_{y_0, z_0}^{NT}; \tilde{n}_{y_0 x_1 z_1} = (n_{y_0} - u_0^{NT}) - \tilde{n}_{NR, z_0}^{CO}; \\ \tilde{n}_{y_1 x_0 z_1} &= u_1^{NT} - \tilde{n}_{y_1, z_0}^{NT}; \tilde{n}_{y_1 x_1 z_1} = (n_{y_1} - u_1^{NT}) - \tilde{n}_{AR, z_0}^{CO}. \end{aligned}$$

For each of these possibly observable datasets  $\{\tilde{n}_{y_k x_j z_i}\}$ , we then find the corresponding generalized likelihood ratio  $G(\{\tilde{n}_{y_k x_j z_i}\}) = q^{H_0}(\{\tilde{n}_{y_k x_j z_i}\})/q^{ML}(\{\tilde{n}_{y_k x_j z_i}\})$ . Note that the parameter space, which we denote as  $(\tilde{\Psi}, \tilde{\mathbf{T}}(\tilde{\Psi}))$ , is specific to each dataset  $\{\tilde{n}_{y_k x_j z_i}\}$ , and differs from the parameter space for the actually observed data  $(\Psi, \mathbf{T}(\Psi))$ .

Given a fixed value of the nuisance parameter  $\psi$ , the corresponding  $\psi$ -specific p-value is then the total probability under  $H_0$  of all datasets  $\{\tilde{n}_{y_k x_j z_i}\}$  with generalized likelihood ratios  $G(\{\tilde{n}_{y_k x_j z_i}\})$  that are at least as extreme as that for the observed data  $\{n_{y_k x_j z_i}\}$ :

$$p^{H_0}(\{n_{y_k x_j z_i}\}; \psi) = \sum_{\substack{\{\tilde{n}_{y_k x_j z_i}\}: \\ G(\{\tilde{n}_{y_k x_j z_i}\}) \leq G(\{n_{y_k x_j z_i}\})}} \Pr(\{\tilde{n}_{y_k x_j z_i}\} \mid \psi, H_0).$$

Since each fixed value of  $\psi$  corresponds to a different number of Compliers and hence a different instance of the null hypothesis, we may report the maximum among all the  $\psi$ -specific p-values as the p-value from our significance test:

$$p^{H_0}(\{n_{y_k x_j z_i}\}) \equiv \max_{\psi \in \Psi} p^{H_0}(\{n_{y_k x_j z_i}\}; \psi). \quad (22)$$

The probability of obtaining a value of the test statistic as extreme as the one observed (21) will never be larger than  $p^{H_0}(\{n_{y_k x_j z_i}\})$ , irrespective of the number of Compliers in the population, and is thus a valid frequentist p-value.

## 5 APPLICATIONS

### 5.1 PSYCHOLOGY DATA EXAMPLE

For the observed dataset  $\{n_{y_k x_j z_i}\}$  in the motivating example from Table 3, the largest probability under  $H_0$  is  $q^{H_0}(\{n_{y_k x_j z_i}\}) = 1 \times 10^{-4}$ ; the maximum likelihood without the constraint of no Compliers who were Helped or Hurt in the population is  $q^{ML}(\{n_{y_k x_j z_i}\}) = 2.3 \times 10^{-3}$ . The generalized likelihood ratio (21) for this dataset is:  $G(\{n_{y_k x_j z_i}\}) = \frac{q^{H_0}(\{n_{y_k x_j z_i}\})}{q^{ML}(\{n_{y_k x_j z_i}\})} = 0.052$ .

There were  $1296 = (23 + 1) \times (53 + 1)$  possible values of the nuisance parameter  $\psi$ , and the maximum among all the  $\psi$ -specific p-values is:

$$p^{H_0}(\{n_{y_k x_j z_i}\}) \equiv \max_{\psi \in \Psi} \{p^{H_0}(\{n_{y_k x_j z_i}\}; \psi)\} = 0.0137.$$

### 5.2 LIPID DATA EXAMPLE

For the observed dataset  $\{n_{y_k x_j z_i}\}$  in the motivating example from Table 4, the largest probability under  $H_0$  is  $q^{H_0}(\{n_{y_k x_j z_i}\}) = 6 \times 10^{-23}$ ; the maximum likelihood without the constraint of no Compliers who were Helped or Hurt in the population is  $q^{ML}(\{n_{y_k x_j z_i}\}) = 0.0019$ . The generalized likelihood ratio (21) for this dataset is then:

$$G(\{n_{y_k x_j z_i}\}) = \frac{q^{H_0}(\{n_{y_k x_j z_i}\})}{q^{ML}(\{n_{y_k x_j z_i}\})} = 3 \times 10^{-20}.$$

There were  $2385 = (14 + 1) \times (158 + 1)$  possible values of the nuisance parameter  $\psi$ , and the maximum among all the  $\psi$ -specific p-values is:

$$p^{H_0}(\{n_{y_k x_j z_i}\}) \equiv \max_{\psi \in \Psi} \{p^{H_0}(\{n_{y_k x_j z_i}\}; \psi)\} = 2 \times 10^{-21}.$$

In comparison, using a pre-specified value of  $\gamma = 0.01$  gives a p-value of  $0.01 + (1 \times 10^{-21}) \approx 0.01$  [10].

## 6 CONCLUSIONS

We have proposed a finite population significance test of the sharp null hypothesis for Compliers using the generalized likelihood ratio. The resulting p-value may be arbitrarily close to zero and summarizes the strength of evidence against the sharp null hypothesis for Compliers (8).

While our development has assumed that there are no Always Takers, the approach extends to the more general case in which there are also Always Takers. However, this would increase the dimension of the nuisance parameter and hence the size of the nuisance parameter space, such that finding the generalized likelihood ratio test statistic would be computationally more intensive. For example, even when the sharp null hypothesis for Compliers (8) holds, the number of Compliers in the  $z = 1$  arm ( $n_{NR, z_1}^{CO}$  and  $n_{AR, z_1}^{CO}$ ) would no longer be point-identified from the observed counts  $n_{y_0 x_1 z_1}$  and  $n_{y_1 x_1 z_1}$  respectively.

## References

- [1] J D Angrist, G W Imbens, and D B Rubin. Identification of causal effects using instrumental variables. *Journal of the American Statistical Association*, 91 (434):444–455, 1996.
- [2] A Balke and J Pearl. Counterfactual probabilities: Computational methods, bounds and applications. In *Proceedings of the Tenth International Conference on Uncertainty in Artificial Intelligence*, pages 46–54. Morgan Kaufmann Publishers Inc., San Francisco, 1994.
- [3] D M Chickering and J Pearl. A clinician’s tool for analyzing non-compliance. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence - Volume 2*, pages 1269–1276. AAAI Press, 1996.
- [4] B Efron and D Feldman. Compliance as an explanatory variable in clinical trials. *Journal of the American Statistical Association*, 86(413):9–17, 1991.
- [5] R A Fisher. *The design of experiments*. Oliver & Boyd, 1935.
- [6] D Heckerman and R Shachter. Decision-theoretic foundations for causal reasoning. *Journal of Artificial Intelligence Research*, 3:405–430, 1995.
- [7] G W Imbens and P R Rosenbaum. Robust, accurate confidence intervals with a weak instrument: quarter of birth and education. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 168(1):109–126, 2005.
- [8] G W Imbens and D B Rubin. Bayesian inference for causal effects in randomized experiments with non-compliance. *Ann. Statist.*, 25(1):305–327, 1997.
- [9] N L Johnson and S Kotz. *Discrete distributions*. Houghton Mifflin, Boston, 1969.
- [10] W W Loh and T S Richardson. A finite population test of the sharp null hypothesis for compliers. *UAI Workshop on Approaches to Causal Structure Learning, 15 July, Bellevue, Washington*, 2013.
- [11] T L Nolen and M G Hudgens. Randomization-based inference within principal strata. *Journal of the American Statistical Association*, 106(494):581–593, 2011.
- [12] W Oberhofer and H Kaufmann. Maximum likelihood estimation of a multivariate hypergeometric distribution. *Sankhya: The Indian Journal of Statistics, Series B (1960-2002)*, 49(2):188–191, 1987.
- [13] J Pearl. *Causality: Models, reasoning, and inference*. Cambridge University Press, Cambridge, second edition, 2009.
- [14] J Pearl. On the consistency rule in causal inference: Axiom, definition, assumption, or theorem? *Epidemiology*, 21(6):872, 2010.
- [15] M D Perlman and L Wu. The emperor’s new tests. *Statistical Science*, 14(4):355–369, 1999.
- [16] K J Rothman, S Greenland, and T L Lash. *Modern Epidemiology*. Wolters Kluwer Health/Lippincott Williams & Wilkins, Philadelphia, 2008.
- [17] D B Rubin. More powerful randomization-based p-values in double-blind trials with non-compliance. *Statistics in Medicine*, 17(3):371–385, 1998.
- [18] A Sommer and S L Zeger. On estimating efficacy from clinical trials. *Statistics in Medicine*, 10(1):45–52, 1991.
- [19] J Sława-Neyman. On the application of probability theory to agricultural experiments. Essay on principles. Section 9. *Statist. Sci.*, 5(4):465–472, 1990. Translated from the Polish and edited by D M Dąbrowska and T P Speed.

---

# Structure Learning Constrained by Node-Specific Degree Distribution

---

Jianzhu Ma<sup>1</sup>, Qingming Tang<sup>1</sup>, Sheng Wang, Feng Zhao, Jinbo Xu  
Toyota Technological Institute at Chicago  
Chicago, IL - 60637  
{majianzhu, qmtang, wangsheng, fzhaoh, j3xu}@ttic.edu

## Abstract

We consider the problem of learning the structure of a Markov Random Field (MRF) when a node-specific degree distribution is provided. The problem setting is inspired by protein contact map (i.e., graph) prediction in which the contact number (i.e., degree) of an individual residue (i.e., node) can be predicted without knowing the contact graph. We formulate this problem using maximum pseudo-likelihood plus a node-specific  $\ell_1$  regularization derived from the predicted degree distribution. Intuitively, when a node have  $k$  predicted edges, we dynamically reduce the regularization coefficients of the  $k$  most possible edges to promote their occurrence. We then optimize the objective function using ADMM and an Iterative Maximum Cost Bipartite Matching algorithm. Our experimental results show that using degree distribution as a constraint may lead to significant performance gain when the predicted degree has good accuracy.

## 1. INTRODUCTION

Structure learning of a Markov Random Field (MRF) is an important problem and has been applied to many real-world problems which require study of conditional independence between a set of objects. For example, structure learning has been used to derive gene expression or regulatory network from gene expression levels [1, 6, 32, 36] and predict the contact map of a protein from a multiple sequence alignment of a protein family [9, 15, 16, 23, 24]. Two major approaches and their variants have been studied to learn the structure of a graphical model from data: Gaussian Graphical Model (GGM) [11] and maximum pseudo-likelihood [29]. Since many real-world

structures are usually sparse,  $\ell_1$  regularization is usually added to the objective function to generate a sparse structure. Empirical studies indicate that the pseudo-likelihood approach may have better prediction accuracy and is also more efficient than GGM, by dropping the Gaussian distribution assumption.

In real-world applications, the underlying structure (or graph) usually has some special properties and must satisfy some topological constraints. For example, a gene expression network is scale-free. A protein contact graph must satisfy some geometric constraints, e.g., the degree of each node is upper bounded by a constant and also depends on the properties of its corresponding amino acid. Only a few structure-learning algorithms take into consideration topological constraints of the underlying graph, which can be used to reduce the feasible solution set of the problem. From another perspective, predicted graphs without considering these constraints might contain conflicts and are physically infeasible. A predicted contact graph violating the above-mentioned geometric constraints may not correspond to a feasible protein structure. Several papers [12, 17, 30, 35] have considered some very general topological constraints describing the global properties of a graph to improve structure learning. However, these non-specific topological constraints do not help very much in practice. The reason may be that they are too loose for some nodes (graphs) and too restrictive for others and thus, the overall performance gain is limited.

The problem addressed by this paper is inspired by protein contact graph prediction. A protein sequence consists of a string of amino acids (also called residues). In nature, a protein sequence folds into a specific 3D shape to function properly. Two residues are defined to form a contact if they are close (distance  $\leq 8$  Å) in the 3D space. Therefore, we can use a contact map to model a protein 3D structure. Predicting inter-residue contacts from sequence is an important and challenging problem. Recent studies [22, 24, 25, 27] indicate that predicted inter-residue contacts could be used as a valuable constraint to improve the folding of some proteins. Baker group [16] shows that one correct long-range contact for every 12 amino acids (AAs) in a

---

<sup>1</sup> The first two authors contribute equally to the paper.

protein allows for accurate topology-level protein folding. Recent breakthroughs [9, 15, 24] apply Gaussian Graphical Model and maximum pseudo-likelihood to formulate protein contact prediction as a structure learning problem. In these formulations, a protein sequence is viewed as a sample generated from a Markov Random Field (MRF), in which an MRF node represents one AA (also called residue) and an edge indicates a contact (i.e., strong interaction) between two AAs.

Without knowing the actual contact graph of a protein, we can use a supervised learning method to predict the number of contacts (i.e., degree) of an AA from sequence information. In particular, we use  $2L$  different linear-chain  $2^{\text{nd}}$ -order Conditional Neural Fields (CNFs) [28] to predict the degree distribution for a protein of length  $L$ . A CNF is an integration of neural networks and Conditional Random Field (CRF) [20]. CNF models the relationship between the label at each node and input features by neural networks and also correlation among neighboring labels. Therefore, CNF can capture the complex relationship between node labels and features as well as the dependency between node labels. The predicted node degree distribution is then used as a regularization to help improve individual contact prediction.

## 2. RELATED WORK

To our best knowledge, there are very few published work that uses node-specific degree distribution to help with structure learning of MRFs. Motivated by the observation that many social and biological networks follow a power-law degree distribution [2, 5, 14], [21] proposed a novel non-convex reweighted  $\ell_1$  regularization by using a log surrogate to approximate the power-law distribution. The basic idea is to reduce the regularization coefficients for hub nodes (i.e., nodes with a large degree) to promote their occurrence. A convex variant of this work was developed in [7], resulting in further performance improvement. This work modeled the structure learning problem as a set-function optimization problem and approximated it by Lovasz extension [4]. The resultant objective function is another kind of reweighted  $\ell_1$  regularization. Although these methods result in a graph following a power-law degree distribution, their accuracy of the predicted edges is not much better than the simple  $\ell_1$  regularization. A very recent work [37] obtained much better accuracy by making use of a reweighted  $\ell_1$  regularization accounting for not only global degree distribution, but also the estimated degree of an individual node and the relative strength of all the edges of the same node.

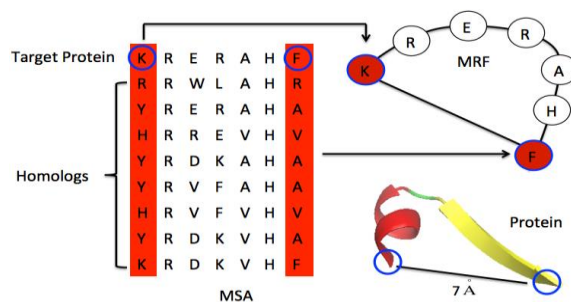
Other work such as [10] takes into consideration eigenvector centrality constraints and triangle-shaped local motifs of the graph, which are properties of gene regulatory networks and protein-protein interaction networks. [33] presented a convex formulation that uses a group-sparsity penalty on the rows and columns of the data precision matrix, effectively selecting which nodes

connect with all the other nodes or no nodes at all. This formulation also results in a graph following a power-law distribution.

## 3. METHOD

### 3.1 NOTATIONS AND PRELIMINARIES

Given a protein sequence, we can run PSI-BLAST [3] to find its sequence homologs (i.e., proteins in the same family) and build a multiple sequence alignment (MSA) of homologs. By examining this MSA, we can identify evolution and co-evolution patterns in a protein family. By co-evolution, we mean the evolution of one AA is strongly impacted by the other. As shown in Figure 1, the AAs in the two red MSA columns are co-evolved. It has been observed that two co-evolved residues are likely to form a contact in the 3D space since they strongly interact with each other.



**Figure 1.** Two coevolved AAs (in red columns) may form a contact in 3D space.

We can use Markov Random Fields (MRF) to model the MSA and infer inter-residue contacts by structure learning of the MRF. In the MRF model, a node represents one MSA column and an edge represents correlation between two MSA columns. Let  $X = \{X_1, X_2, \dots, X_L\}$  be a protein sequence where  $X_i$  represents amino acid type (or gap) at column  $i$ . Let  $R$  denote the number of protein sequences (or rows) in the MSA. Let  $X_{ir}$  denote the amino acid type observed at row  $r$  ( $1 \leq r \leq R$ ) and column  $i$  ( $1 \leq i \leq L$ ). The probability of observing  $X$  can be defined as follows.

$$P(X) = \prod_{r=1}^R \frac{\exp(\sum_{i=1}^L b_i(X_{ir}) + \sum_{i<j} w_{ij}(X_{ir}, X_{jr}))}{Z} \quad (1)$$

Here  $b_i$  and  $w_{ij}$  denote the unary and binary potential functions for nodes  $i$  and  $j$ , respectively.  $Z$  is the partition function, summing over all the possible label combinations. If nodes  $i$  and  $j$  share an edge in the graph, they are correlated given all the other nodes, indicating that their corresponding AAs form a contact and interact with each other in the 3D space. Therefore, the contact number of one AA corresponds to the node degree in the graph. Both training and inference by maximizing (1) over a general graph are NP-hard. Pseudo-likelihood approximation [9, 29] is proposed to



deal with this. Substituting the original likelihood function, we have,

$$P(X_i) = \prod_{r=1}^R P(X_{i,r} | X_{\setminus i,r}) \quad (2)$$

$$= \prod_{r=1}^R \frac{\exp(b_i(X_{i,r}) + \sum_{j=1, j \neq i}^L w_{i,j}(X_{i,r}, X_{j,r}))}{\sum_{X_{i,r}} \exp(b_i(X_{i,r}) + \sum_{j=1, j \neq i}^L w_{i,j}(X_{i,r}, X_{j,r}))}$$

Each binary potential function  $w_{ij}$  is a  $21 \times 21$  matrix. We can estimate all  $w_{ij}$  by maximizing (2). We can use  $\sum_{a,b=1}^{20} |w_{i,j}(a,b)|$  to measure the interaction strength between two nodes  $i$  and  $j$ . A pair of nodes with strong interaction is predicted to share an edge or form a contact.

### 3.2 NODE-SPECIFIC DEGREE REGULARIZATION

In this section we introduce how to add a node-specific degree distribution as a prior to the above pseudo-likelihood function. Let  $P_i(k)$  be the predicted probability of node  $i$  having  $k$  contacts. Let  $W_{ij} = \sum_{a,b=1}^{20} |w_{i,j}(a,b)|$ , which indicates the interaction strength between two AAs  $i$  and  $j$ . Given a  $i$ , we exclude  $W_{ii}$  and denote the  $t$ -th largest  $W_{ij}$  as  $W_{i,(t)}$ . We use the following penalty term  $\Omega_i$  for AA  $i$ .

$$\Omega_i = \sum_{k=1}^{L-1} P_i(k) \left( -\sum_{t=1}^k W_{i,(t)} + \sum_{t=k+1}^{L-1} W_{i,(t)} \right) \quad (3)$$

Eq. (3) implies that if the degree of AA  $i$  is  $k$ , its  $k$  largest  $W_{i,(1)}, W_{i,(2)}, \dots, W_{i,(k)}$  shall be big and the remaining  $W_{i,(k+1)}, W_{i,(k+2)}, \dots$  shall be very small. The outer summation ranges from 1 to  $L-1$  since the contact number is less than  $L$ . Regrouping Eq. (3) by each  $W_{i,(k)}$  we have,

$$\Omega_i = \sum_{k=1}^{L-1} g_{i,k} W_{i,(k)} \quad (4)$$

where  $g_{i,k} = -\sum_{t=k}^{L-1} P_i(t) + \sum_{t=k+1}^{L-1} P_i(t)$ . Notice that the coefficient  $g_{i,k}$  for  $W_{i,(k)}$  can be negative, which will lead the optimization problem to be unbounded, so empirically we add a constant  $\beta$  (0.2 by default) to each  $g_{i,k}$  to make it positive. Figure 2 shows two examples. For node  $i$ , its degree is most likely to be either 1 or 3, so the coefficient for the largest interaction strength  $W_{i,(1)}$  should be reduced. For node  $j$ , its degree is most likely to be zero, so all the coefficients are increased to drive its interaction strengths to zero.

The reweighted  $\ell_1$  regularized pseudo-likelihood function is defined as,

$$\min_W L(W) + \sum_{i=1}^L \Omega_i(W) \quad (5)$$

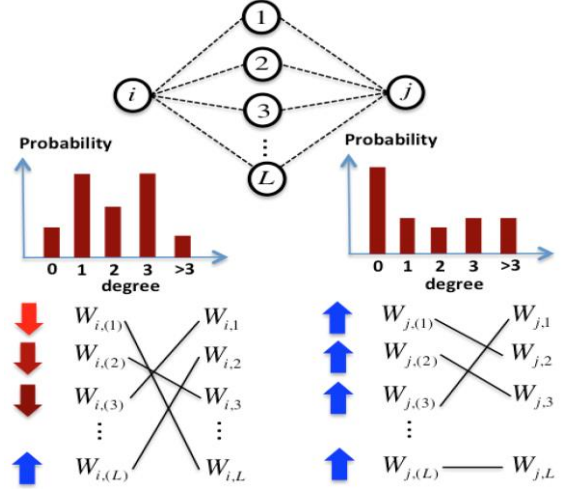
Where  $L(W)$  is the negative log of Eq. (2) and  $\Omega_i$  is a special  $\ell_1$  penalty defined in Eq. (4). To optimize (5), we

use Alternating Direction Method of Multipliers (ADMM) [13] to separate the pseudo-likelihood function and the  $\ell_1$  penalty term. ADMM alternatively solves the following three sub-problems.

$$Z^{n+1} = \operatorname{argmin}_Z L(Z) + \frac{\rho}{2} \|Z - W^n + U^n\|_2^2 \quad (6)$$

$$W^{n+1} = \operatorname{argmin}_W \sum_{i=1}^L \Omega_i(W) + \frac{\rho}{2} \|Z^n - W + U^n\|_2^2 \quad (7)$$

$$U^{n+1} = U^n + Z^{n+1} - W^{n+1} \quad (8)$$



**Figure 2.** Node-specific degree-based  $\ell_1$  penalty. This figure shows how the regularizers are different based on the predicted degree distribution. The X-axis is the probability and the Y-axis is the degree. The colorful arrows represent the change of coefficient for the  $\ell_1$ -norm. The warmer the color the smaller the value is. Down arrows represent negative value and up arrows represent positive value.

where  $\rho > 0$  is a fixed step-size parameter (we used  $\rho = 0.01$ ) and  $U$  is the dual variable passing information between sub-problems (6) and (7). Problem (6) can be solved using conjugate gradient decent. Since the order of  $W_{i,*}$  for each  $i$  is unknown in problem (7), it is challenging to solve (7). We need to consider their order so that  $g_{i,k}$  can be used to weight the  $k$ -th largest  $W_{i,(k)}$ . Let  $M = Z + U$  and  $g_{i,k} = g_{i,k}/\rho$ . We may further divide problem (7) into  $L$  sub-problems; the  $i$ -th sub-problem is as follows.

$$\min_{W_i} \frac{1}{2} \|W_i - |M_i|\|_2^2 + \Omega_i(W_i) \quad (9)$$

To solve problem (9), we need a mapping between original  $\{W_{i,k}\}$  and  $\{g_{i,k}\}$ , as different mappings lead to different optimization problems. For a given mapping, we need to minimize the corresponding function subject to the constraints provided by the mapping. We want the mapping with the smallest optimal value. However, there are an exponential number of mappings between  $\{W_{i,k}\}$  and  $\{g_{i,k}\}$ , which makes it impossible to enumerate all the possible mappings. To make it easy, we assume  $W_{i,(k)} >$

$W_{i,(k+1)}$  for all  $k$ . That is, we only want to find a solution satisfying this condition. Next we will introduce how to use an Iterative Maximum Cost Bipartite Matching algorithm to solve the relaxation problem.

### 3.2.1 Iterative Maximum Cost Bipartite Matching Algorithm

**Theorem 1.** Let  $W_{i,(1)} > W_{i,(2)} > \dots > W_{i,(L-1)}$  be the ranking of  $\{W_{i,k}\}$ . The optimal solution  $W_{i,(k)}^* = \sum_{a,b=1}^{20} |w_{i,(k)}^*(a,b)|$  of problem (9) always has the form,

$$|w_{i,(k)}^*(a,b)| = \max\{|M_{i,(k)}(a,b)| - g_{i,k}, 0\} \quad (10)$$

**Proof.** By taking the sub-gradient with respect to each  $|w_{i,(k)}(a,b)|$  and setting it to zero we obtain Eq. (10). If the optimal solution  $|w_{i,(k)}^*(a,b)|$  of (9) does not satisfy Eq. (10), we can always decrease the objective function by adding or subtracting a small constant to (10) so that  $W_{i,(k-1)} > W_{i,(k)} > W_{i,(k+1)}$  still holds. This contradicts with our assumption that  $|w_{i,(k)}^*(a,b)|$  is the optimal solution.  $\square$

Based on Theorem 1, given a ranking of  $\{W_{i,k}\}$ , we can substitute  $\{W_{i,k}\}$  of (9) by Eq. (10) to obtain the below equation.

$$\sum_{k=1}^{L-1} \sum_{a,b=1}^{20} M_{i,(k)}(a,b)^2 - w_{i,(k)}^*(a,b)^2 \quad (11)$$

Now we need to minimize (11). Notice that the summation of all the  $M_{i,(k)}(a,b)^2$  is a constant, so minimizing (11) is equivalent to the following optimization problem.

$$\max \sum_{k=1}^{L-1} \sum_{a,b=1}^{20} w_{i,(k)}^*(a,b)^2 \quad (12)$$

Substituting (9) into (12) and considering the constraints of the mapping, we have the following optimization problem.

$$\max \sum_{k=1}^{L-1} \sum_{a,b=1}^{20} \max\{|M_{i,(k)}(a,b)| - g_{i,k}, 0\}^2 \quad (13)$$

$$s.t. \quad \forall k \quad W_{i,(k)}^* > W_{i,(k+1)}^*$$

where  $W_{i,(k)}^* = \sum_{a,b=1}^{20} \max\{|M_{i,(k)}(a,b)| - g_{i,k}, 0\}$ . Note that in this problem we are looking for a one-to-one matching between  $M_{i,(k)}$  and  $g_{i,k}$ , which can be modeled as an Integer Linear Problem (ILP) defined on variable  $E = \{e_{k,l}\}$  as follows,

$$\begin{aligned} \max \sum_{k,l} \theta_{k,l} e_{k,l} + \sum_{k \neq q,l} \theta_{k,q,l,l+1} e_{k,l} e_{q,l+1} \quad (14) \\ s.t. \quad \forall k, l \quad \sum_k e_{k,l} = 1, \sum_l e_{k,l} = 1 \end{aligned}$$

Here  $e_{k,l} = 1$  if  $M_{i,k}$  is assigned to  $g_{i,l}$ ; otherwise 0. Let  $\Lambda_{i,k,l}(a,b) = \max\{|M_{i,k}(a,b)| - g(i,l), 0\}$ , then each  $\theta_{k,l}$  and  $\theta_{k,q,l,l+1}$  can be computed as follows.

$$\begin{aligned} \theta_{k,l} &= \sum_{a,b=1}^{20} \Lambda_{i,k,l}(a,b)^2 \quad (15) \\ \theta_{k,q,l,l+1} &= \begin{cases} 0 & \sum_{a,b=1}^{20} \Lambda_{i,k,l}(a,b) > \sum_{a,b=1}^{20} \Lambda_{i,q,l+1}(a,b) \\ -\infty & \text{otherwise} \end{cases} \end{aligned}$$

Each  $\theta_{k,l}$  reflects the preference of mapping  $M_{i,k}$  to  $g_{i,l}$  while  $\theta_{k,q,l,l+1}$  reflects the constraints to be satisfied. With these definitions, we can use the ADMM algorithm by introducing an auxiliary variable  $v_{k,l}$  for each  $e_{k,l}$  and solving (14) using the following iterative procedure,

$$V^{n+1} = \operatorname{argmin}_V \sum_{q,l} C_{q,l} v_{q,l} \quad (16)$$

$$E^{n+1} = \operatorname{argmin}_E \sum_{k,l} D_{k,l} e_{k,l} \quad (17)$$

$$\eta^{n+1} = \eta^n + (E^{n+1} - V^{n+1}) \quad (18)$$

Each  $C_{q,l}$  and  $D_{k,l}$  can be computed as,

$$C_{q,l} = -\eta_{q,l} + (\sum_{k\{k \neq q\}} \theta_{k,q,l-1,l} e_{k,l-1}) + \gamma e_{q,l} \quad (19)$$

$$D_{k,l} = \theta_{k,l} + \sum_{q\{q \neq k\}} \theta_{k,q,l,l+1} v_{q,l+1} + \eta_{k,l} + \gamma v_{k,l} \quad (20)$$

Here  $\gamma > 0$  is a fixed step-size parameter (we used  $\gamma = 0.5$ ) and  $\eta$  (we used 0.1) is the dual variable passing information between sub-problems (16) and (17). Both (16) and (17) can be viewed as a bipartite matching problem, which can be solved by the Hungarian algorithm [19].

Using the above algorithm (steps 16-18), we can find a permutation of  $\{W_{i,k}\}$ ;  $|w_{i,(k)}^*(a,b)|$  is then given by (10). Notice that in order to minimize (9), if  $|w_{i,(k)}^*(a,b)| \neq 0$ , then  $w_{i,(k)}^*(a,b)$  and  $M_{i,k}(a,b)$  should have the same sign. The final solution of  $w_{i,(k)}^*(a,b)$  is therefore given by,

$$\begin{aligned} w_{i,(k)}^*(a,b) \quad (21) \\ = \begin{cases} M_{i,(k)}(a,b) - \operatorname{sign}(M_{i,(k)}(a,b)) g_{i,k} & |M_{i,(k)}(a,b)| > g_{i,k} \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

### 3.3 ESTIMATE NODE-SPECIFIC DEGREE DISTRIBUTION

Here we introduce how to predict the degree distribution of each node. Notice that in a protein sequentially-adjacent AAs must be close in the 3D space so their corresponding MRF nodes are connected. We introduce a concept called Partial Contact Number (PCN) denoted by  $B_{i,j}$  for each node pair  $i$  and  $j$ , which is defined as the number of edges formed by node  $i$  with nodes  $i+1, i+2, \dots, j$  (if  $j > i$ ) or nodes  $i-1, i-2, \dots, j$  (if  $j < i$ ). Each  $B_{i,j}$  has 15 labels indicating the degree from 0 to 13 and  $\geq 14$ . We set the maximum degree to 14 since the contact number of an AA is upper bounded by a small constant. Since  $B_{i,j}$  is correlated with its nearest neighbors  $B_{i,j-1}$  and  $B_{i,j+1}$ , we apply a Conditional Neural Field (CNF) to predict  $B_{i,j}$ . Each node  $i$  is associated with two 2<sup>nd</sup>-order CNFs, as

shown in Figure 3. A protein with  $L$  nodes (AAs) has  $2L$  different 2<sup>nd</sup>-order CNFs.

Let  $F_{i,j}$  denote the feature vector extracted from two AAs  $i$  and  $j$ , we use one CNF is to estimate  $P(B_{i,i+6} \sim B_{i,L} | F_{i,i+6} \sim F_{i,L})$  and the other for  $P(B_{i,1} \sim B_{i,i-6} | F_{i,i+6} \sim F_{i,L})$ . We ignore very short-range contacts (sequence distance  $< 6$ ) as they are less informative for structure prediction. We train CNFs by maximum likelihood. We use a  $\ell_2$  regularization to avoid over-fitting and 5-fold cross validation to choose the hyper parameters. Since CNF is non-convex, we train it starting from 5 different initial solutions and pick the best one. See [28] for more details of CNF.

After training the CNF models, we calculate the marginal probabilities  $P(B_{i,j})$  using the standard forward-backward algorithm [20] independently on each CNF. Finally, we calculate the probability of node  $i$  having degree  $K$  as follows.

$$\sum_{K_1+K_2=K} (P(B_{i,1} = K_1) + P(B_{i,L} = K_2)) \quad (22)$$

## 4. EXPERIMENTS

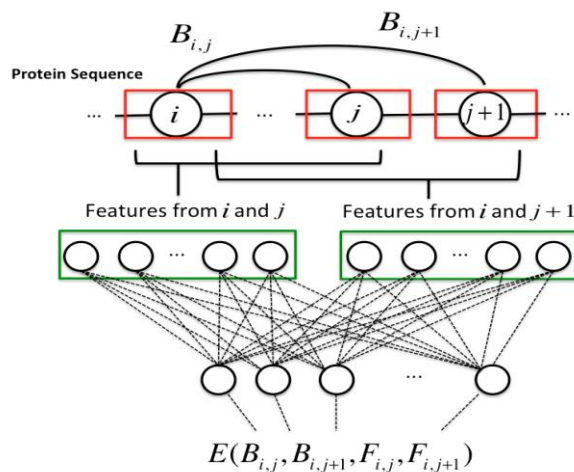
### 4.1 TRAINING AND TEST DATA

We use a subset of the PDB25 dataset, generated by the PISCES server [34], to train and validate our CNF models. Any two proteins in this dataset share  $< 25\%$  sequence identity. In total we used 3118 proteins with length between 40 and 500, among which 3/4 are randomly chosen for training and the remaining 1/4 for validation. To test the performance, we evaluate our results on CASP10 [18] and CASP11 [26] datasets. We rule out short proteins with fewer than 70 amino acids since they have relatively low contact number prediction accuracy. This leads to 109 test proteins in the CASP10 set and 99 proteins in the CASP11 dataset. We use the CASP official domain boundary definition for each test protein. For each test protein, we run PSI-BLAST [3] with 5 iterations and E-value 0.001 to generate sequence profile, from which we extract  $F_{i,j}$ . All the native structures of our training and validation proteins are solved before CASP10 and CASP11 and do not share high sequence identify with the CASP test proteins.

### 4.2 EVALUATION CRITERIA AND PROGRAMS TO COMPARE

Depending on the sequence distance (i.e., the number of AAs between the two ends of a contact along the protein sequence), we divide contacts into 3 categories: [6,12) for short-range contacts, [12,24) for medium-range contacts and  $\geq 24$  for long-range contacts. Generally speaking, medium- and long-range contacts are more important for structure prediction, but more challenging to predict. We

evaluate only top  $L/5$ ,  $L/10$ , and  $L/2$  predicted contacts. The accuracy is calculated as the percentage of the correctly predicted contacts. The ground truth is calculated from the experimental structure. When more predicted contacts are evaluated, the difference among methods becomes smaller since it is more likely to pick a native contact by chance. We compare our method to three other structure learning methods: PSICOV [15], plmDCA [9], and CCMpred [31]. PSICOV uses Graphical Lasso for contact prediction while plmDCA and CCMpred use maximum pseudo-likelihood with  $\ell_2$  regularization. These programs are run with their default parameters. When node-specific degree distribution is no used, our method is exactly the same as CCMpred, so we can calculate performance gain by examining the improvement of our method over CCMpred.



**Figure 3.** A Conditional Neural Field model for the prediction of Partial Contact Numbers.

### 4.3 PRE-PROCESSING AND POST-PROCESSING

We employ the same pre- and post-processing procedures as plmDCA and CCMpred to ensure our comparison with them is fair. To reduce the impact of redundant sequences, we apply the same sequence weighting method as plmDCA. In particular, duplicate sequences are removed and MSA columns containing more than 90% of gaps are deleted. The sequence is weighted using a threshold of 62% sequence identity. Similar to plmDCA and CCMpred, average-product correction (APC) [8] is applied to post-process predicted contacts.

### 4.4 PERFORMANCE

#### 4.4.1 Overall Performance

As shown in Tables 1 and 2, on both CASP10 and CASP11 test proteins, our method significantly outperforms the others in terms of the accuracy of the top  $L/10$ ,  $L/5$  and  $L/2$  predicted contacts. plmDCA and CCMpred achieve better

results than PSICOV because they drop the Gaussian distribution assumption. Our method differs from plmDCA and CCMpred in that we use a separate  $\ell_1$  regularization term for every pair of AAs instead of a universal regularization on all the AA pairs.

**Table 1.** Contact prediction accuracy on the 109 CASP10 targets

|            | Short-range |             |             | Medium-range |             |             | Long-range  |             |             |
|------------|-------------|-------------|-------------|--------------|-------------|-------------|-------------|-------------|-------------|
|            | L/10        | L/5         | L/2         | L/10         | L/5         | L/2         | L/10        | L/5         | L/2         |
| Our Method | <b>0.32</b> | <b>0.33</b> | <b>0.19</b> | <b>0.39</b>  | <b>0.34</b> | <b>0.28</b> | <b>0.37</b> | <b>0.34</b> | <b>0.25</b> |
| PSICOV     | 0.23        | 0.19        | 0.14        | 0.31         | 0.26        | 0.19        | 0.28        | 0.23        | 0.17        |
| plmDCA     | 0.26        | 0.22        | 0.15        | 0.34         | 0.29        | 0.21        | 0.33        | 0.28        | 0.21        |
| CCMpred    | 0.28        | 0.29        | 0.16        | 0.36         | 0.30        | 0.22        | 0.33        | 0.30        | 0.22        |

**Table 2.** Contact prediction accuracy on the 99 CASP11 targets

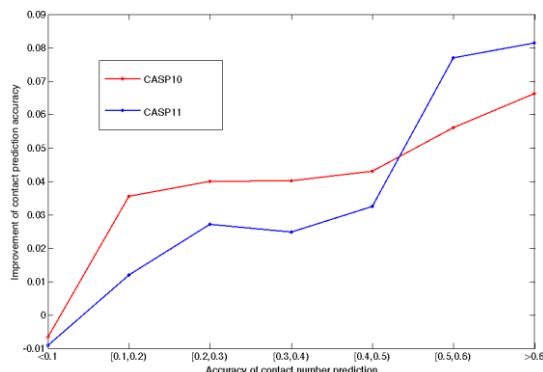
|            | Short-range |             |             | Medium-range |             |             | Long-range  |             |             |
|------------|-------------|-------------|-------------|--------------|-------------|-------------|-------------|-------------|-------------|
|            | L/10        | L/5         | L/2         | L/10         | L/5         | L/2         | L/10        | L/5         | L/2         |
| Our method | <b>0.25</b> | <b>0.22</b> | <b>0.15</b> | <b>0.27</b>  | <b>0.22</b> | <b>0.16</b> | <b>0.28</b> | <b>0.25</b> | <b>0.19</b> |
| PSICOV     | 0.19        | 0.14        | 0.11        | 0.20         | 0.16        | 0.12        | 0.20        | 0.17        | 0.13        |
| plmDCA     | 0.19        | 0.14        | 0.11        | 0.21         | 0.17        | 0.13        | 0.23        | 0.23        | 0.17        |
| CCMpred    | 0.21        | 0.17        | 0.12        | 0.24         | 0.19        | 0.13        | 0.24        | 0.22        | 0.17        |

#### 4.4.2 Impact of Predicted Contact Number Distribution

First we evaluate the accuracy of contact number prediction. The contact number is predicted by picking the label with the maximum marginal probability computed by (22). The 15-label accuracy calculated on the CASP10 and CASP11 datasets are both 0.30 while random guess (i.e., predicting all the labels to be the one with the largest background probability) is 0.21. The average Pearson correlations between the ground truth and our prediction are 0.71 and 0.74, respectively. In addition, we can predict small- or large-valued contact number labels very accurately. These two properties help suppress the contacts of those AAs with very few contacts from showing up in the final prediction and thus, decrease the false positives.

Now we evaluate the impact of contact number prediction on individual contact prediction. We compare our method (i.e., predicted contact number used) with CCMpred (i.e., predicted contact number not used) in terms of the accuracy of the top  $L/10$  predicted long-range contacts. The top  $L/10$  predicted contacts cover only a small number of AAs, so for each protein we only calculate the contact number accuracy on the AAs covered by the top  $L/10$  predicted contacts. We group the test proteins into seven bins according to their accuracy of predicted contact number. For each bin we calculate the average accuracy improvement of individual contact prediction by our method over CCMpred. As shown in Figure 4, the improvement is positively correlated with the accuracy of contact number prediction on both CASP10 and CASP11 datasets. That is, the more accurately we can predict the

contact number, the more performance gain can be obtained for individual contact prediction. In particular, when the accuracy of contact number prediction is low ( $<0.1$ ), our method cannot improve individual contact prediction accuracy because the predicted contact number has too much noise. When the accuracy of contact number prediction is above 0.5, the performance gain from the predicted contact number information is large ( $\geq 0.05$ ). This implies that our method makes a good use of predicted contact number information.



**Figure 4.** Relationship between top  $L/10$  long-range contact prediction accuracy gain and the accuracy of contact number prediction. The accuracy gain is calculated as the performance difference between our method and CCMpred.

#### 4.4.3 Case Study

Here we use two specific examples to further demonstrate the strength of our method. In particular, we want to study how the predicted contact number distribution helps with individual contact prediction. One example is a CASP10 target T0758 (PDB ID 4RM7). The other is a CASP11 target T0813 (PDB ID 4WJI). They have 366 and 302 AAs, respectively, and 9572 and 4177 similar sequences. As shown in Tables 3 and 4, our method significantly outperforms the others by at least 0.2 on the top  $L/10$  long-range contact predictions. plmDCA and CCMpred have similar results since they use the same loss function. PSICOV yields relatively low performance on T0813, most likely attributing to the default sparsity setting being too aggressive.

**Table 3.** Contact prediction accuracy of T0758 (4RM7)

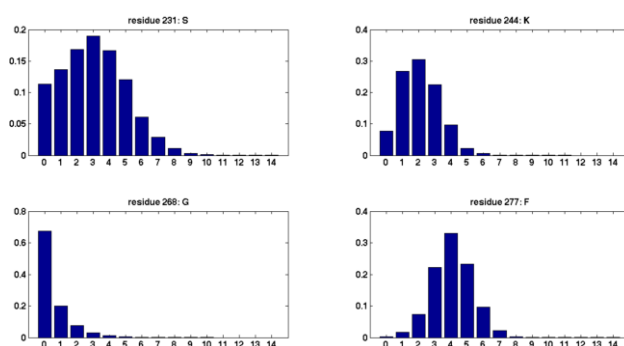
|            | Short-range |             |             | Medium-range |             |             | Long-range  |             |             |
|------------|-------------|-------------|-------------|--------------|-------------|-------------|-------------|-------------|-------------|
|            | L/10        | L/5         | L/2         | L/10         | L/5         | L/2         | L/10        | L/5         | L/2         |
| Our method | <b>0.62</b> | <b>0.39</b> | <b>0.26</b> | <b>0.67</b>  | <b>0.55</b> | <b>0.28</b> | <b>0.76</b> | <b>0.64</b> | <b>0.50</b> |
| PSICOV     | 0.50        | 0.31        | 0.19        | 0.55         | 0.41        | 0.19        | 0.50        | 0.47        | 0.40        |
| plmDCA     | 0.44        | 0.30        | 0.20        | 0.61         | 0.42        | 0.26        | 0.56        | 0.56        | 0.45        |
| CCMpred    | 0.42        | 0.32        | 0.19        | 0.64         | 0.56        | 0.30        | 0.53        | 0.48        | 0.45        |

**Table 4.** Contact prediction accuracy of T0813 (4WJI)

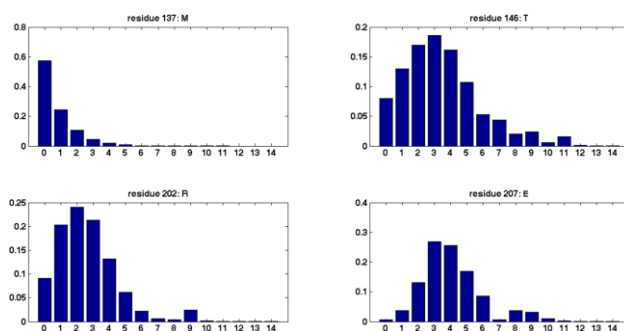
|  | Short-range |     |     | Medium-range |     |     | Long-range |     |     |
|--|-------------|-----|-----|--------------|-----|-----|------------|-----|-----|
|  | L/10        | L/5 | L/2 | L/10         | L/5 | L/2 | L/10       | L/5 | L/2 |

| Our method | 0.43 | 0.32 | 0.16 | 0.60 | 0.44 | 0.25 | 0.73 | 0.60 | 0.50 |
|------------|------|------|------|------|------|------|------|------|------|
| PSICOV     | 0.30 | 0.20 | 0.11 | 0.43 | 0.28 | 0.19 | 0.50 | 0.40 | 0.31 |
| plmDCA     | 0.37 | 0.27 | 0.14 | 0.57 | 0.40 | 0.20 | 0.53 | 0.47 | 0.40 |
| CCMpred    | 0.37 | 0.28 | 0.13 | 0.53 | 0.42 | 0.23 | 0.47 | 0.50 | 0.45 |

Now for each target we examine two AA pairs not in contact. Our method can correctly predict that they are not in contact, but the other three methods predict they are in contact. Figures 5 and 6 show the predicted contact number distributions for the eight AAs of the two targets. Most of these AAs are predicted to have very few contacts. Especially the 137th AA of T0758 and the 268th AA of T0813; the predicted probability of the contact number being zero are both over 0.5, which means all the parameters associated with these two AAs shall be forced to zero.

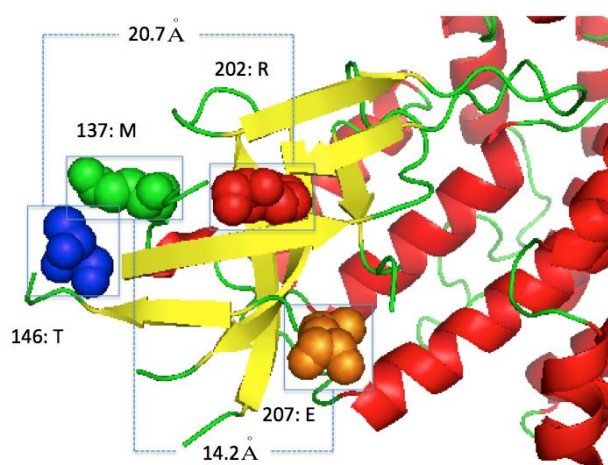


**Figure 5.** Predicted contact number distributions of 4 AAs of T0758 (4RM7) shown in Figure 7.

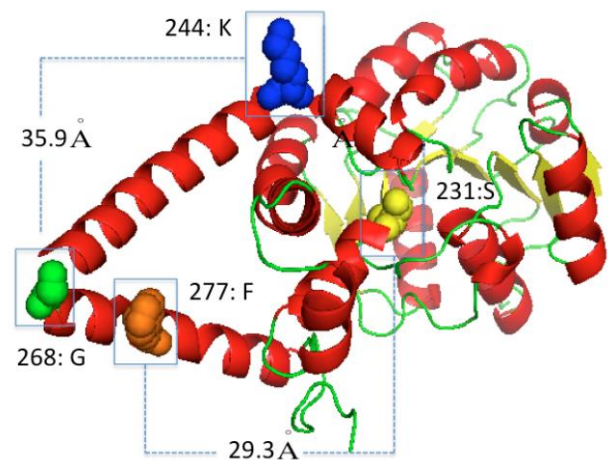


**Figure 6.** Predicted contact number distributions of 4 AAs of T0813 (4WJI) shown in Figure 8.

As shown in Figures 7 and 8, these two AAs are exposed at the protein surface and they do not form any long-range contacts. Similarly, for the other 6 AAs, the mass of predicted contact number distribution are all concentrated around 3 or 4, which means our model most likely can only allow 3 or 4 contacts for each AA. The top predicted contacts of these AAs are all short- and medium-range. That is why our method does not predict any long-range contacts for these AAs.



**Figure 7.** Two long-range false positives predicted by PSICOV, plmDCA and CCMpred for Target T0758 (4RM7): one false contact between the 146th AA and 202nd AA and the other between the 137th AA and 207th AA. Their true distances are 20.7 Å and 14.2 Å, respectively.



**Figure 8.** Two false positives predicted by PSICOV, plmDCA and CCMpred for T0813 (4WJI): one false contact between the 244th AA and 268th AA and the other between the 231st AA and 277th AA. Their true distances are 35.9 Å and 29.3 Å, respectively.

## 5. DISCUSSION AND FUTURE WORK

We have presented a new structure learning method that can make use of the predicted node-specific degree distribution to improve prediction accuracy of edges. The predicted degree distribution is used as a kind of soft topological constraints to restrict the solution space and avoid “unreasonable” predictions. Experimental results show that by using the degree distribution we can significantly improve protein contact prediction over current state-of-the-art structure learning methods.

From a computational perspective, our method provides a new framework to integrate orthogonal information into structure learning. That is, we first use supervised learning to learn node-specific local topological

constraints and then add it as a prior to learn the whole network structure. In many real-world applications, the connections of the graph are more or less influenced by the properties of nodes, so a node-specific degree distribution can be learned from local features without knowing the whole network structure. The contact number prediction is a very challenging supervised learning problem. In this work, we use multiple linear-chain graphical models to circumvent the difficulty of training and inference on loopy graphs. In the future, we will extend CNF by adding a deep learning module to further improve it.

For protein contact prediction, adding AA-specific topological constraint is only our first step. We are considering other AA- and segment-specific topological constraints, such as some geometric constraints imposed by a single secondary structure segment, two correlated secondary structure segments, or even the global structure of a protein.

### Acknowledgements

We thank Payman Yadollahpour for useful discussions. This work is financially supported by the NIH grant R01GM089753 the NSF grant DBI-1262603 (to J.X.) and the NSF CAREER award CCF-1149811 (to J.X.). The authors are also grateful to the computational resources provided by the University of Chicago RCC.

### References

- Ahmed A, Song L, Xing EP (2008) Time-varying networks: Recovering temporally rewiring genetic networks during the life cycle of *Drosophila melanogaster*. arXiv preprint arXiv:0901.0138
- Albert R, Barabási A-L (2002) Statistical mechanics of complex networks. *Reviews of modern physics* 74:47
- Altschul SF, Madden TL, Schäffer AA et al. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic acids research* 25:3389-3402
- Bach FR (2010) Structured sparsity-inducing norms through submodular functions. In: *Advances in Neural Information Processing Systems*. p 118-126
- Barabási A-L, Albert R (1999) Emergence of scaling in random networks. *science* 286:509-512
- Celik S, Logsdon B, Lee S-I (2014) Efficient Dimensionality Reduction for High-Dimensional Network Estimation. In: *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*. p 1953-1961
- Defazio A, Caetano TS (2012) A convex formulation for learning scale-free networks via submodular relaxation. In: *Advances in Neural Information Processing Systems*. p 1250-1258
- Dunn SD, Wahl LM, Gloor GB (2008) Mutual information without the influence of phylogeny or entropy dramatically improves residue contact prediction. *Bioinformatics* 24:333-340
- Ekeberg M, Lökqvist C, Lan Y et al. (2013) Improved contact prediction in proteins: using pseudolikelihoods to infer Potts models. *Physical Review E* 87:012707
- Fiori M, MuséP, Sapiro G (2012) Topology constraints in graphical models. In: *Advances in Neural Information Processing Systems*. p 791-799
- Friedman J, Hastie T, Tibshirani R (2008) Sparse inverse covariance estimation with the graphical lasso. *Biostatistics* 9:432-441
- Hayat S, Elofsson A (2012) BOCTOPUS: improved topology prediction of transmembrane  $\beta$  barrel proteins. *Bioinformatics* 28:516-522
- Hestenes MR (1969) Multiplier and gradient methods. *Journal of optimization theory and applications* 4:303-320
- Jeong H, Mason SP, Barabási A-L et al. (2001) Lethality and centrality in protein networks. *Nature* 411:41-42
- Jones DT, Buchan DW, Cozzetto D et al. (2012) PSICOV: precise structural contact prediction using sparse inverse covariance estimation on large multiple sequence alignments. *Bioinformatics* 28:184-190
- Kamisetty H, Ovchinnikov S, Baker D (2013) Assessing the utility of coevolution-based residue-residue contact predictions in a sequence- and structure-rich era. *Proceedings of the National Academy of Sciences* 110:15674-15679
- Klepeis J, Floudas C (2003) ASTRO-FOLD: a combinatorial and global optimization framework for ab initio prediction of three-dimensional structures of proteins from the amino acid sequence. *Biophysical Journal* 85:2119-2146
- Kryshtafovych A, Barbato A, Fidelis K et al. (2014) Assessment of the assessment: evaluation of the model quality estimates in CASP10. *Proteins: Structure, Function, and Bioinformatics* 82:112-126
- Kuhn HW (1955) The Hungarian method for the assignment problem. *Naval research logistics quarterly* 2:83-97
- Lafferty J, McCallum A, Pereira FC (2001) Conditional random fields: Probabilistic models for segmenting and labeling sequence data.
- Liu Q, Ihler AT (2011) Learning scale free networks by reweighted  $l_1$  regularization. In: *International Conference on Artificial Intelligence and Statistics*. p 40-48
- Ma J, Wang S, Wang Z et al. (2014) MRAlign: protein homology detection through alignment of Markov random fields. *PLoS computational biology* 10:e1003500

23. Ma J, Wang S, Xu J (2013) Protein contact prediction by joint evolutionary coupling analysis across multiple families. arXiv preprint arXiv:1312.2988
24. Marks DS, Colwell LJ, Sheridan R et al. (2011) Protein 3D structure computed from evolutionary sequence variation. *PLoS one* 6:e28766
25. Michel M, Hayat S, Skwark MJ et al. (2014) PconsFold: improved contact predictions improve protein models. *Bioinformatics* 30:i482-i488
26. Moutl J, Fidelis K, Kryshchuk A et al. (2014) Critical assessment of methods of protein structure prediction (CASP)—round x. *Proteins: Structure, Function, and Bioinformatics* 82:1-6
27. Nugent T, Jones DT (2012) Accurate de novo structure prediction of large transmembrane protein domains using fragment-assembly and correlated mutation analysis. *Proceedings of the National Academy of Sciences* 109:E1540-E1547
28. Peng J, Bo L, Xu J (2009) Conditional neural fields. In: *Advances in neural information processing systems*. p 1419-1427
29. Ravikumar P, Wainwright MJ, Lafferty JD (2010) High-dimensional Ising model selection using  $\ell_1$ -regularized logistic regression. *The Annals of Statistics* 38:1287-1319
30. Savojardo C, Fariselli P, Martelli PL et al. (2013) BCov: a method for predicting  $\beta$ -sheet topology using sparse inverse covariance estimation and integer programming. *Bioinformatics*:btt555
31. Seemayer S, Gruber M, Söding J (2014) CCMpred—fast and precise prediction of protein residue–residue contacts from correlated mutations. *Bioinformatics* 30:3128-3130
32. Sharan R, Ulitsky I, Shamir R (2007) Network - based prediction of protein function. *Molecular systems biology* 3
33. Tan KM, London P, Mohan K et al. (2014) Learning graphical models with hubs. *The Journal of Machine Learning Research* 15:3297-3331
34. Wang G, Dunbrack RL (2003) PISCES: a protein sequence culling server. *Bioinformatics* 19:1589-1591
35. Wang Z, Xu J (2013) Predicting protein contact map using evolutionary and physical constraints by integer programming. *Bioinformatics* 29:i266-i273
36. Wei Z, Li H (2007) A Markov random field model for network-based analysis of genomic data. *Bioinformatics* 23:1537-1544
37. Xu J, Com G Learning Scale-Free Networks by Dynamic Node-Specific Degree Prior.

---

# Active Search and Bandits on Graphs Using Sigma-Optimality

---

**Yifei Ma**  
Machine Learning Department  
Carnegie Mellon University  
yifeim@cs.cmu.edu

**Tzu-Kuo Huang\***  
Microsoft Research  
tkhuang@microsoft.com

**Jeff Schneider**  
Robotics Institute  
Carnegie Mellon University  
schneide@cs.cmu.edu

## Abstract

Many modern information access problems involve highly complex patterns that cannot be handled by traditional keyword based search. Active Search is an emerging paradigm that helps users quickly find relevant information by efficiently collecting and learning from user feedback. We consider active search on graphs, where the nodes represent the set of instances users want to search over and the edges encode pairwise similarity among the instances. Existing active search algorithms are either short of theoretical guarantees or inadequate for graph data. Motivated by recent advances in active learning on graphs, namely the  $\Sigma$ -optimality selection criterion, we propose new active search algorithms suitable for graphs with theoretical guarantees and demonstrate their effectiveness on several real-world datasets.

We relate our active search setting to multi-armed bandits whose rewards are binary values indicating search hits or misses and arms cannot be pulled more than once. We also discussed theoretical guarantees for applying  $\Sigma$ -optimality as the exploration term for bandits on graphs.<sup>1</sup>

## 1 INTRODUCTION

As the world gets increasingly digitized and electronically recorded, how to quickly identify relevant pieces of information becomes a major issue. Internet search engines are

---

\* Part of this work was done while the author was with Carnegie Mellon University.

<sup>1</sup>An earlier version of this paper included results on bandit cumulative regrets with improved rates (originally Section 4.2.2). These results depended on proof strategies from Contal et al. (2014) (originally in Appendix C) which were found to be incorrect. Therefore, these results have been removed in the current version of the paper.

an integral part of modern life, serving as a probe into the diverse, complex and expanding space of human digital traces. Despite being successful in many information retrieval tasks, the keyword-based query mechanism in most search engines may fall short when targets are characterized by complex patterns or signatures beyond keywords. For example, financial transactions associated with illegal activities bear signatures involving multiple factors such as time, location, occupation of the account owner, etc. In the investigation of organizational misconduct, such as the Enron scandal, the important leads or evidences, oftentimes buried in a sea of diverse electronic and paper trails, usually involve information exchange among key individuals and their relationship. To fully understand the users' intent in these cases, keyword-based search may serve as a good starting point, but is certainly far from completing the task.

Such needs of more general search paradigms have recently motivated several efforts Garnett et al. (2012); Wang et al. (2013); Vanchinathan et al. (2013), most of which are related to the Active Search framework proposed by Garnett et al. (2012). It is an interactive search mechanism that begins with a full set of instances without supervision and a given task/keyword-specific similarity measure between these instances. Based on the similarity measure and an optional initial set of suggestions from the user, an algorithm figures out what instances the user should examine next and presents it to the user, who then decides whether the presented instance is relevant or not. Upon receiving this feedback, the algorithm updates its search strategy accordingly and selects the next instance to present. The loop continues until the user quits, and the goal is to maximize the total number of relevant instances found.

As one can see, Active Search has close connections to some well-studied machine learning paradigms. At a first glance, Active Learning (Settles, 2010) seems the most related because they both ask for user feedback incrementally and adaptively. However, Active Learning aims at improving generalization performances with as few label queries as possible, while Active Search is evaluated by how many relevant instances it found along the way, and therefore



must carefully balance exploitation and exploration. This trade-off relates Active Search to stochastic optimization in the Multi-Armed Bandit setting (Robbins, 1985; Dani et al., 2008; Kleinberg et al., 2008; Bubeck et al., 2009), where the goal is to find the maximum of an unknown function using as few function evaluations as possible. However, Active Search deviates from this setting in that it selects instances *without replacement* and is competing with the best *subset* of instances rather than the single best.

We investigate Active Search when the instances are represented by the nodes on a graph whose edges encode pairwise similarity among the instances. For a toy example, please see Figure 1. Many real-world datasets are of this type, such as web pages, citation networks, and e-mail correspondences. For data that are not naturally represented as graphs, a graph representation based on pairwise similarity can still be beneficial because it may reveal useful manifold structures (Tenenbaum et al., 2000; Belkin and Niyogi, 2001). Existing active search approaches (Wang et al., 2013; Garnett et al., 2012; Vanchinathan et al., 2013) either lack theoretical guarantees or ignore certain graph properties, thereby degrading empirical performances. By drawing ideas from recent advances in active learning on graphs (Ma et al., 2013), we proposed new active search algorithms with theoretical guarantees, and empirically demonstrate their advantages over existing methods. In particular, our new exploration criteria, motivated by  $\Sigma$ -optimality criterion (Ma et al., 2013) for active learning on graphs, favor nodes with not only high uncertainty, but also high influence on the other nodes.

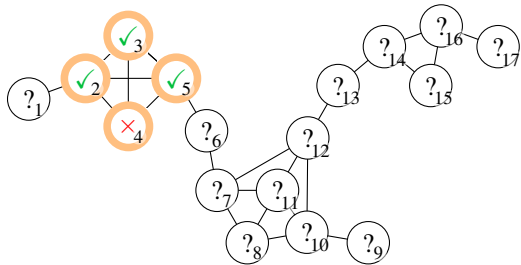


Figure 1: A toy examples for active search where the goals are “ $\checkmark$ ” nodes. Suppose the yellow nodes are observed in previous rounds, which node should be searched next?

The rest of the paper is organized as follows. We describe related work in Section 2, and introduce the problem setup in Section 3. We then present our new methods in Section 4 along with theoretical guarantees, followed by experimental results in Section 5.

## 2 RELATED WORK

Wang et al. (2013) proposed an active search algorithm for graphs, building on label propagation and semi-supervised learning using Gaussian random fields (Zhu

et al., 2003a,b). Despite decent empirical performances, this approach does not have any theoretical guarantee. Vanchinathan et al. (2013) proposed a Gaussian-Process (GP) based algorithm, GP-SELECT, for sequentially selecting instances with high user scores or ratings (rewards). This algorithm extends the popular GP-UCB algorithm (Cox and John, 1997; Auer, 2003) for stochastic optimization and inherits nice theoretical guarantees (Srinivas et al., 2012). When applied to graphs, however, it tends to select nodes at the periphery of the graph because they have large predictive variances, leading to large exploration factors in the GP-UCB selection rule. Yet the rewards of these nodes reveal little information about the reward distribution over the whole graph.

Similar issues have been observed in active learning on graphs as well. In their experiments, Ma et al. (2013) found that selection rules based on mutual information gain (Krause et al., 2008), which is closely related to per-node predictive variances, usually end up selecting nodes at the periphery of a graph. Ji and Han (2012) proposed a selection criterion based on one-step lookahead decrease of the average variance of all remaining nodes, which effectively considers not only the predictive variance of the search node itself, but also its covariances with all remaining nodes. This criterion corresponds to standard V-optimality in experiment design. Ma et al. (2013) further improved the state of the art by using the  $\Sigma$ -optimality criterion, which demonstrates greater robustness against outliers and better empirical performances than V-optimality. Motivated by these recent advances, we propose new active search algorithms that combine GP-UCB with  $\Sigma$ -optimality.

Valko et al. (2014) considered bandit problems where arms correspond to nodes on a graph and the reward is a smooth function over the graph. Their algorithm can be viewed as a special case of GP-UCB with a kernel defined by the inverse of a graph Laplacian (augmented with an identity matrix). To analyze the performance of their UCB-style algorithm, they propose the notion of *effective dimension* of a graph, which can be viewed as a measure of the spectral decay of the kernel, thereby determining the performance of the algorithm (Srinivas et al., 2012). We also use the effective dimension to analyze our proposed methods.

## 3 PROBLEM SETUP

The database where active search is performed is given as a graph  $\mathcal{G}$  with known structure (edge connections). The edge connections are nonnegative and we use  $\mathbf{A}$  to represent the adjacency matrix of  $\mathcal{G}$ , such that  $A_{ij} \geq 0, \forall i, j$ . Let  $\mathcal{V} = \{v_1, \dots, v_n\}$  denote the set of all nodes in  $\mathcal{G}$ . From  $\mathbf{A}$  we can derive a graph Laplacian matrix,  $\mathcal{L} = \mathbf{D} - \mathbf{A}$ , where  $\mathbf{D} = \text{diag}(\mathbf{A} \cdot \mathbf{1}) = \text{diag}(\text{deg}(v_1), \dots, \text{deg}(v_n))$ .

Every node  $v$  in our graph holds one reward value we denote as  $f(v)$ , indicating whether the node is the search tar-

get. The reward is unknown at first and can be revealed only when it is queried explicitly. For mathematical benefits, we relax the reward to be a real value and introduce a Gaussian noise to its observation, as

$$y(v) = f(v) + \epsilon, \text{ where } \epsilon \sim \mathcal{N}(0, \sigma_n^2). \quad (1)$$

Similar to bandit problems, querying a node also means collecting the true reward of that node. Our goal is to design a query strategy, which interactively generates a query sequence  $\mathbf{v}_t = (v_1, \dots, v_t)^\top$  without any repeated selections, in order to maximize the cumulative reward

$$F_T = \sum_{t=1}^T f(v_t). \quad (2)$$

The cumulative reward is always upper-bounded by the optimal strategy with full knowledge of the true rewards on all the nodes. Let  $\mathbf{v}_t^* = (v_1^*, \dots, v_t^*)$  to be the optimal query sequence (without repeated selections), our analysis in Theorem 2 (Section 4.2) bounds the cumulative regret between our strategy and the optimal strategy,

$$R_T = \sum_{t=1}^T f(v_t^*) - f(v_t). \quad (3)$$

The above characterizes an active search problem, provided that the values of  $f(v)$  are binary and the sequences  $\mathbf{v}_t$  and  $\mathbf{v}_t^*$  do not allow repeated selections. Otherwise, the above can also model a multi-armed bandit problem if we relax  $f(v)$  to be real and  $\mathbf{v}_t$  and  $\mathbf{v}_t^*$  to allow repeated selections. In fact, our formulation discusses them together, providing analysis to the slightly more rigorous active search modeling except that  $f(v)$  is relaxed to real values.

In our notations, bold letters indicate vectors or matrices, while light letters without subscripts mean functions and light letters with subscripts represent scalars or specific elements.  $t$ ,  $\tau$ , and  $T$  are time indices, which when applied as subscripts, always mean the selection or model at that time step. Other letters as subscripts, such as  $i, j, n$ , always mean the natural indices.

### 3.1 GAUSSIAN RANDOM FIELD PRIOR

A key assumption in this work is that the reward values, or the target labels, are constrained by the graph structure in a non-trivial way. Otherwise, the input graph provides little information about the reward function, making active search extremely difficult. More specifically, we assume that the reward values of all the nodes in the graph, collectively denoted as a vector  $\mathbf{f} \in \mathbb{R}^N$ , are random variables

distributed jointly as

$$\log p(\mathbf{f}) \simeq - \sum_{i=1}^N \sum_{j=1}^N \frac{A_{ij}(f_i - f_j)^2}{2} - \sum_{j=1}^N \frac{\omega_0(f_j - \mu_0)^2}{2},$$

$$\text{i.e., } \mathbf{f} \sim \mathcal{N}\left(\boldsymbol{\mu}_0 = \mu_0 \cdot \mathbf{1}, \mathbf{C}_0 = (\boldsymbol{\mathcal{L}} + \omega_0 \mathbf{I})^{-1}\right), \quad (4)$$

where  $\mu_0$  is a prior mean, and  $\omega_0 > 0$  is a regularization parameter. According to this probabilistic model, it is more likely for connected nodes to share similar values than not. Define the initial covariance matrix as,  $\mathbf{C}_0 = (\boldsymbol{\mathcal{L}} + \omega_0 \mathbf{I})^{-1}$ , and denote  $\tilde{\boldsymbol{\mathcal{L}}}_0 = \boldsymbol{\mathcal{L}} + \omega_0 \mathbf{I}$ . The above prior model is also known as **Gaussian random fields (GRFs)**.

### 3.2 POSTERIOR INFERENCE

Assume the nature draws one sample from the prior model, (4), and we use query observations, (1), to converge to that particular draw by performing posterior inference conditioned on the history,

$$\mathcal{H}_t = \{(v_\tau, y_\tau)\}_{\tau=1}^t = \{\mathbf{v}_t, \mathbf{y}_t\},$$

which allows us to update the posterior distribution as,

$$\log p(\mathbf{f} | \mathcal{H}_t) \simeq -\frac{1}{2}(\mathbf{f} - \boldsymbol{\mu}_0)^\top \tilde{\boldsymbol{\mathcal{L}}}_0(\mathbf{f} - \boldsymbol{\mu}_0) - \sum_{\tau=1}^t \frac{(y_\tau - f_{v_\tau})^2}{2\sigma_n^2}.$$

Notice that the prior distribution and likelihood model form Gaussian conjugate pairs. Denote the posterior distribution as,  $\mathbf{f} | \mathcal{H}_t \sim \mathcal{N}(\boldsymbol{\mu}_t, \mathbf{C}_t)$ . To some readers, it is easier to express  $\boldsymbol{\mu}_t$  and  $\mathbf{C}_t$  using the prior *precision* matrix, as

$$\boldsymbol{\mu}_t = \mathbf{C}_t \left( \tilde{\boldsymbol{\mathcal{L}}}_0 \boldsymbol{\mu}_0 + \sum_{\tau=1}^t \frac{y_\tau \mathbf{e}_{v_\tau}}{\sigma_n^2} \right), \quad \mathbf{C}_t^{-1} = \tilde{\boldsymbol{\mathcal{L}}}_0 + \frac{1}{\sigma_n^2} \mathbf{H}_t \quad (5)$$

where  $\mathbf{e}_{v_\tau} = (0, \dots, 0, 1, 0, \dots, 0)^\top$  is an indicator vector of index  $v_\tau$  and  $\mathbf{H}_t$  is a diagonal matrix of index counts from  $\mathbf{v}_t$ , whose  $k$ th diagonal element is  $\sum_{\tau=1}^t e_{v_\tau}(v_k)$ .

However, for convenience in later descriptions and to connect to **Gaussian Process (GP)** literature (Rasmussen and Williams, 2006), we also use the prior *covariance* matrix to express the posterior distribution, as,

$$\mu_t(v) = \mu_0(v) + \mathbf{c}_{\mathbf{v}_t v}^\top (\mathbf{C}_{\mathbf{v}_t \mathbf{v}_t} + \sigma_n^2 \mathbf{I})^{-1} (\mathbf{y}_t - \boldsymbol{\mu}_{\mathbf{v}_t}),$$

$$C_t(v, v') = C_0(v, v') - \mathbf{c}_{\mathbf{v}_t v}^\top (\mathbf{C}_{\mathbf{v}_t \mathbf{v}_t} + \sigma_n^2 \mathbf{I})^{-1} \mathbf{c}_{\mathbf{v}_t v'}, \quad (6)$$

where the matrices can all be defined in terms of the prior:

$$\mathbf{c}_{\mathbf{v}_t v} = (C_0(v_1, v), \dots, C_0(v_t, v))^\top$$

$$\mathbf{C}_{\mathbf{v}_t \mathbf{v}_t} = (C_0(v_\tau, v_{\tau'}))_{\tau, \tau'=1}^t$$

$$\boldsymbol{\mu}_{\mathbf{v}_t} = (\mu_0(v_1), \dots, \mu_0(v_t))^\top.$$

The above update rules also applies to any time interval that starts with  $t_0$ , by replacing prior models (variables with subscript “0”) with the model at time  $t_0$ .

Define simple notations for correlation coefficients and standard deviations from the covariance matrix,  $C_t(v, v') = \rho_t(v, v')\sigma_t(v)\sigma_t(v')$ , which implies that  $\sigma_t^2(v) = C_t(v, v)$ . Define  $\mathbf{c}_t(v)$  to be the column of  $\mathbf{C}_t$  corresponding to node  $v$ .

## 4 METHOD

---

**Algorithm 1** GP-SOPT and its variants

---

**input**  $\mu_0, \mathbf{A}, \omega_0, \sigma_n, \alpha_t, T$ ; if warm start,  $\{v_\tau, y(v_\tau)\}_{\tau=1}^{t_0}$   
 1: Obtain initial  $\mathcal{N}(\mu_0, \mathbf{C}_0)$  // (4)  
 2: **for**  $t = t_0, \dots, T - 1$ , **do**  
 3: Update to posterior  $\mathcal{N}(\mu_t, \mathbf{C}_t)$  // (6)  
 4:  $v_{t+1} \leftarrow \arg \max_{v \in V \setminus S_t} \mu_t(v) + \alpha_{t+1}s_t(v)$  // (9.a, 9.b, or 9.c)  
 5: Observe  $y(v_{t+1})$ ; include  $S_{t+1} \leftarrow S_t \cup \{v_{t+1}\}$   
 6: **end for**  
**output**  $S_T$ .

---

Our proposed active search algorithms are described in Algorithm 1. They resemble general exploration-exploitation style algorithms with GPs. Here we focus on binary functions that assign value 1 to relevant or target nodes, and 0 to all other nodes. At iteration  $t + 1$ , Algorithm 1 selects the next node to query based on a deterministic selection rule of the form:

$$\arg \max_{v \in V \setminus S_t} \mu_t(v) + \alpha_{t+1} \cdot s_t(v), \quad (7)$$

where  $\mu_t(v)$  is the usual exploitation term and  $s_t(v)$  encourages exploration, with the two being balanced by a possibly iteration-dependent parameter  $\alpha_{t+1} > 0$ .

Examples from existing literature like the popular GP-UCB algorithm and its extension to Active Search, GP-SELECT (Vanchinathan et al., 2013), amount to setting  $s_t(v)^2 = \sigma_t(v)^2$ , the posterior (as well as predictive) variance of the reward value at node  $v$ . Although this is a very reasonable choice in many situations, it may lead to undesirable exploration behaviors on graphs. Under our model assumption, low-degree nodes, which usually lie at the periphery of a graph, tend to have high predictive variances. Direct applications of GP-UCB may result in the selection of many such outliers, which fail to reveal much information about the reward values of most other nodes at the core of the graph (Figure 2(a)).

Intuitively, a good exploration criterion should favor nodes that have high influences on other parts of the graph. That is, the knowledge of the function values at these nodes should reveal a lot about the function values at other nodes. Under our model assumption, this principle naturally connects with the predictive covariances of a node with others. Research in active learning on graphs has already made use of predictive covariances to construct better selection rules.

Ji and Han (2012) proposed to select nodes based on their sums of squares of predictive covariances with other nodes, which is derived from the minimization of squared prediction error, known as V-optimality in experiment design. Ma et al. (2013) observed that V-optimality can still be undesirably sensitive to outliers and used  $\Sigma$ -**optimality criterion** instead, which by itself selects a set of nodes  $\mathbf{v}_t$  to minimize the following Bayes *survey risk* on the posterior model after the selection,

$$\mathcal{R}_{t|\mathbf{v}_t}^\Sigma = \mathbb{E} \left( \sum_{v' \in \mathcal{V}} f_{t|\mathbf{v}_t}(v') - \sum_{v' \in \mathcal{V}} \mu_{t|\mathbf{v}_t}(v') \right)^2 = \mathbf{1}^\top \mathbf{C}_{t|\mathbf{v}_t} \mathbf{1}.$$

For active search, we use this criterion in a greedy sequential selection manner for exploration scoring, as

$$\begin{aligned} s_t(v) &= \sqrt{\mathcal{R}_{t|S_t}^\Sigma - \mathcal{R}_{t+1|S_t \cup \{v\}}^\Sigma} = \frac{\sum_{v'} C_t(v, v')}{\sqrt{C_t(v, v) + \sigma_n^2}} \\ &= \frac{1}{\sqrt{1 + \sigma_n^2/\sigma_t^2(v)}} \cdot \sum_{v' \in \mathcal{V}} \rho_t(v, v')\sigma_t(v'), \end{aligned} \quad (8)$$

where the second equality is easily derived from (6). If we ignore  $\sigma_n$  (set it to 0), the  $\Sigma$ -optimality criterion (8) considers the sum of a node's correlation times standard deviation of all nodes on the graph. High score nodes by this criterion are likely to provide rich information for exploration.

We propose three exploitation-exploration style algorithms with exploration criteria motivated by  $\Sigma$ -optimality, which are *vanilla*  $\Sigma$ -*optimality* and its two variants with an additional parameter  $k$  that we will describe next. All algorithms select the next node to query by the general rule (7), but use different exploration terms:

**GP-SOPT (Vanilla  $\Sigma$ -Optimality):**

$$s_t(v) = \frac{1}{\sqrt{1 + \sigma_n^2/\sigma_t^2(v)}} \cdot \sum_{v' \in \mathcal{V}} \rho_t(v, v')\sigma_t(v'). \quad (9.a)$$

**GP-SOPT.TT (Thresholded Total Covariance):**

$$s_t(v) = \min \left( k\sigma_t(v), \sum_{v' \in \mathcal{V}} \rho_t(v, v')\sigma_t(v') \right). \quad (9.b)$$

**GP-SOPT.TOPK (Top- $k$  Covariance):**

$$s_t(v) := \max_{B \subset \mathcal{V}, |B|=k} \sum_{v' \in B} \rho_t(v, v')\sigma_t(v'). \quad (9.c)$$

As one can see in Figure 2(b), the nodes selected by vanilla GP-SOPT indeed reside in more central parts of the toy graph than the nodes selected by its competitor. In a large graph with many peripheral nodes, we believe that the improved exploration criteria of GP-SOPT and its variants contribute to a better recall rate of search targets in real graphs in Section 5.

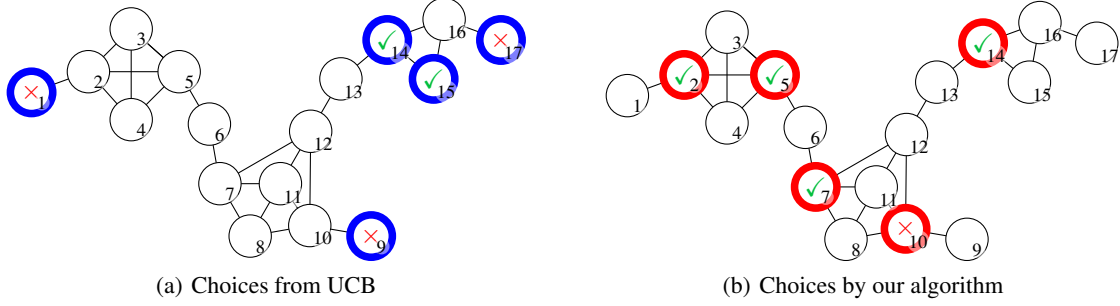


Figure 2: For the toy graph example, choices from (a) direct application of UCB (Vanchinathan et al., 2013; Valko et al., 2014) versus (b) our vanilla GP-SOPT. We observe that our method (b) tends to select more from cluster centers, which helps reduce variance of the unobserved values/rewards, whereas previous literature (a) tends to select the graph periphery.

The reason we propose the latter two variants, (9.b) and (9.c), is to both address proof difficulties and increase practical robustness. By Lemma 3 in Appendix A, we have that  $s_t(v) \geq \sigma_t(v)$  for both criteria, meaning that  $s_t(v)$  maintains the UCB property. Note that the observation noise,  $\sigma_n$ , is also dropped from (9.b) and (9.c). As we will show in our theoretical analysis, we put a threshold in (9.b) against  $k\sigma_t(v)$ , where  $k$  is a tuning parameter, in order to explicitly control the regret of the algorithm. As implied by Lemma 4 in Appendix A, the Top- $k$  Covariance criterion (9.c) is also always upper-bounded by  $k\sigma_t(v)$ .

In the next two subsections we discuss in more details the properties of various exploration criteria, and present our theoretical analysis.

#### 4.1 DISCUSSIONS

Our approach and two other popular criteria, information gain from Srinivas et al. (2012) and V-optimality of Ji and Han (2012), can also be connected by functions of the eigenvalues of the covariance matrix at each iteration.

To see this connection, assume the updated covariance matrix at iteration  $(t + 1)$  has eigen-decomposition  $\mathbf{C}_t = \sum_{j=1}^n \lambda_{t,(j)} \mathbf{q}_{t,(j)} \mathbf{q}_{t,(j)}^\top$ , where  $\boldsymbol{\lambda}_t = (\lambda_{t,(1)}, \dots, \lambda_{t,(n)})^\top$  represents the eigenvalues and  $\{\mathbf{q}_{t,(j)} : j = 1, \dots, n\}$  is the set of corresponding eigenvectors. Assume the eigenvalues are sorted by  $\lambda_{t,(1)} \geq \dots \geq \lambda_{t,(n)} \geq 0$ . We hope to connect  $s_t(v)$  to the following spectral difference,

$$\Delta h_t(v) = h(\boldsymbol{\lambda}_t) - h(\boldsymbol{\lambda}_{t+1|v}) \quad (10)$$

where  $h(\boldsymbol{\lambda}) : \mathbb{R}^n \rightarrow \mathbb{R}$  is a multivariate function defined on the eigenvalues. Further, by the one-step update rule of (6),  $\mathbf{C}_t$  has Loewner order as  $\mathbf{C}_0 \succ \mathbf{C}_1 \succ \dots \succ \mathbf{C}_T \succ \mathbf{0}$ . It is thus often desirable to require  $h(\cdot)$  to be monotone with respect to this ordering, i.e.  $\mathbf{C}_t \succ \mathbf{C}_{t'} \Rightarrow h(\boldsymbol{\lambda}_t) \geq h(\boldsymbol{\lambda}_{t'})$ .

**Case 1.**  $h(\boldsymbol{\lambda}) = \sum_j \log(\lambda_{(j)})$ . Then,  $\Delta h_t(v) = 2\mathcal{I}_t(\mathbf{f}; y(v)) = \log(1 + \frac{\sigma_t^2(v)}{\sigma_n^2})$ , twice the information gain from  $\mathbf{f} \sim \mathcal{N}(\boldsymbol{\mu}_t, \mathbf{C}_t)$  to  $\mathcal{N}(\boldsymbol{\mu}_{t+1|v}, \mathbf{C}_{t+1|v})$ . This metric

is important to **GP-UCB** (Srinivas et al., 2012), which set  $s_t(v) = \sigma_t(v)$  and used the inequality,  $\log(1 + \frac{\sigma_m^2}{\sigma_n^2}) \frac{\sigma_t^2(v)}{\sigma_m^2} \leq \log(1 + \frac{\sigma_t^2(v)}{\sigma_n^2})$ , where  $\sigma_m = \max_{v,t} \sigma_t(v)$ , in its proofs.

**Case 2.**  $h(\boldsymbol{\lambda}) = \sum_j \lambda_{(j)}$  gives  $\Delta h_t(v) = \text{tr}(\mathbf{C}_t) - \text{tr}(\mathbf{C}_{t+1|v}) = \|\mathbf{c}_t(v)\|_2^2 / (\sigma_t^2(v) + \sigma_n^2)$ . For  $\sigma_n = 0$ ,  $\Delta h_t(v)$  is used as the greedy **V-optimal** criterion for design of experiments by Ji and Han (2012).

**Case 3.**  $h(\boldsymbol{\lambda}) = \lambda_{(1)}$  connects to the greedy design for **E-optimality** (Pukelsheim, 1993). To some extent, it is also related to greedy  **$\Sigma$ -Optimality**. First, approximate  $\Delta h_t(v)$  by  $\partial \lambda_{(j)} = \mathbf{q}_{(j)}^\top \partial(\mathbf{C}) \mathbf{q}_{(j)}$  around  $\mathbf{C} = \mathbf{C}_t$ , as

$$\Delta h_t(v) \approx \mathbf{q}_{t,(1)}^\top (\mathbf{C}_t - \mathbf{C}_{t+1|v}) \mathbf{q}_{t,(1)} = \left( \frac{|\mathbf{c}_t(v)^\top \mathbf{q}_{t,(1)}|}{\sqrt{\sigma_t(v)^2 + \sigma_n^2}} \right)^2$$

The above resembles (8) if  $\mathbf{q}_{t,(1)} \propto \mathbf{1}$ , which holds true for  $t = 0$  and  $\omega_0 = 0$  and approximately so for small  $t$ s.

In all these cases, exploration is measured by how much the objective,  $h(\boldsymbol{\lambda}_T)$ , is eventually decreased after  $T$  iterations. Each definition of  $h(\boldsymbol{\lambda}_t)$  aggregates the eigenvalues of the posterior covariance matrices in a different way, which affects the relative importance of large and small eigenvalues. In **Case 1**, since  $\frac{\partial \log(\lambda)}{\partial \lambda} = \frac{1}{\lambda}$ , the same change introduced to a smaller  $\lambda$  will have a relatively larger impact on the objective. Such an effect is not evident in the other two cases. Particularly in **Case 3**, changes to small eigenvalues are ignored unless they become the largest eigenvalue.

Establishing biases to penalize larger eigenvalues more has the benefit of improving global robustness because the posterior marginal variance of every node is upper-bounded by  $\lambda_{t,(1)}$ . Compared with **Cases 2** and **3**, **Case 1** is more sensitive to changes in small eigenvalues, which may be another explanation of **GP-UCB**'s strong tendency to select peripheral nodes, as seen in Figure 4 of Krause et al. (2008) or Figure 1(d) of Gotovos et al. (2013).

Although Algorithm 1 is not built around the concept of functions on eigenvalues, it still establishes strong biases to penalize large eigenvalues in its initial explorations, per

analysis in **Case 3**. Note that  $\Sigma$ -**optimality** achieves a more complex goal than **E-optimality**; exact execution of **E-optimality** may over-simplify the model and select nodes between clusters for separation rather than inside them.

## 4.2 REGRET ANALYSIS

We present an UCB-style analysis for GP-SOPT.TT and GP-SOPT.TOPK. We combine several results on GP optimization (Srinivas et al., 2012; Vanchinathan et al., 2013) and the spectral bandit analysis (Valko et al., 2014). As in these results, our regret bounds depend on the mutual information between  $f$  and the observed values  $\mathbf{y}_S$  at a set  $S$  of nodes:

$$\mathcal{I}(\mathbf{y}_S; f) := H(\mathbf{y}_S) - H(\mathbf{y}_S | f), \quad (11)$$

where  $H(\cdot)$  denotes the entropy. If  $f$  is drawn from a GP with observation noise distributed independently as  $\mathcal{N}(0, \sigma_n)$ , the mutual information has the following analytical form:

$$\mathcal{I}(\mathbf{y}_S; f) = \mathcal{I}(\mathbf{y}_S; f_S) = \frac{1}{2} \log |I + \sigma_n^{-2} \mathbf{C}_{\mathbf{v}_S \mathbf{v}_S}|. \quad (12)$$

Let

$$\gamma_T := \max_{S \in \mathcal{V}, |S|=T} \frac{1}{2} \log |I + \sigma_n^{-2} \mathbf{C}_{\mathbf{v}_S \mathbf{v}_S}|, \quad (13)$$

i.e., the maximum information about  $f$  gained by observing  $T$  function evaluations. The regrets of our algorithms depend on the growth rate of  $\gamma_T$ , which can be linear in  $T$  for arbitrary graphs. However, real-world graphs often possess rich structures, such as clusters or communities, and practical measures of relevance are often highly correlated with these structures, resulting in slowly-growing  $\gamma_T$ . To formalize this intuition, we follow Valko et al. (2014) to consider the *effective dimension*:

$$d_T^* := \max \left\{ i \mid \lambda_i \leq \frac{\sigma_n^{-2} T}{(i-1) \log(1 + \frac{T}{\sigma_n^2 \omega_0})} \right\}, \quad (14)$$

where  $\lambda_i$  is the  $i$ th smallest eigenvalue of  $\tilde{\mathcal{L}}_0$  and  $\lambda_1 = \omega_0$ . The effective dimension is small when the first few  $\lambda_i$ 's are small and the rest increase rapidly, as is often the case for graphs with community or cluster structures. On the contrary, if all the eigenvalues are close to  $\omega_0$ , then  $d_T^*$  may be linear in  $T$ . The following lemma bounds  $\gamma_T$  in terms of  $d_T^*$ :

**Lemma 1.** *Let  $T$  be the total number of rounds. Then*

$$\gamma_T \leq 2d_T^* \log \left( 1 + \frac{T}{\sigma_n^2 \omega_0} \right).$$

*Proof.* By Lemma 7.6 of Srinivas et al. (2012) and the fact that  $\lambda_i^{-1}$  is the  $i$ th largest eigenvalue of the kernel  $\mathbf{C}_0 =$

$\tilde{\mathcal{L}}_0^{-1}$ , we have

$$\gamma_T \leq \max_{\substack{\{m_i\}_{i=1}^T, m_i \geq 0, \\ \sum_{i=1}^T m_i = T}} \sum_{i=1}^T \log \left( 1 + \frac{m_i}{\sigma_n^2 \lambda_i} \right). \quad (15)$$

Then by applying the same argument that proves Lemma 6 of Valko et al. (2014), we obtain the desired result.  $\square$

We will then derive regret bounds in terms of  $\gamma_T$ .

Recall the cumulative regret of an active search algorithm is defined as  $R_T := \sum_{t=1}^T f(v_t^*) - f(v_t)$ , where  $\{v_t\}_{t=1}^T$  is the sequence of unique nodes selected by the algorithm. For the two proposed UCB-style algorithms, GP-SOPT.TT (9.b) and GP-SOPT.TOPK (9.c), we give the following bounds on their cumulative regrets.

**Theorem 2.** *Pick  $\delta \in (0, 1)$ . Assume the vector of true node values,  $\mathbf{f}$ , has bounded quadratic norm,  $\|\mathbf{f}\|_{\tilde{\mathcal{L}}_0} = \sqrt{\mathbf{f}^\top \tilde{\mathcal{L}}_0 \mathbf{f}} \leq B$ ,<sup>2</sup> and the observation noise  $\epsilon_t$  is zero-mean conditioned on the past and is bounded by  $\sigma_n$  almost surely. If GP-SOPT.TT and GP-SOPT.TOPK use GRF prior (4) with zero-mean and graph Laplacian  $\tilde{\mathcal{L}}_0$ , the observation noise model  $\mathcal{N}(0, \sigma_n^2)$ , and  $\alpha_t := \sqrt{2B + 300\gamma_t \log^3(t/\delta)}$ , then their cumulative regrets will satisfy*

$$\Pr(\{R_T \leq k\sqrt{c_1 T \alpha_T \gamma_T} \ \forall T \geq 1\}) \geq 1 - \delta,$$

where the randomness is over the observation noise and  $c_1 := \frac{8/\omega_0}{\log(1 + \sigma_n^{-2})}$ . This implies that with high probability,

$$R_T = O(k\sqrt{T}(B\sqrt{d_T^*} + d_T^*)).$$

This result is easily derived from the regret analysis of the GP-SELECT algorithm proposed by Vanchinathan et al. (2013) because the exploration terms used by GP-SOPT.TT and GP-SOPT.TOPK both satisfy  $\sigma_t(v) \leq s_t(v) \leq k\sigma_t(v)$ , thereby maintaining the UCB property. Although our regret bound is  $k$  times worse than the GP-SELECT bound, the actual regret tends to behave more favorably as we observe in our experiments that after a few tens of rounds,  $s_t(v)$  becomes smaller than  $k\sigma_t(v)$  for almost all unqueried nodes, and the two proposed algorithm usually outperforms GP-SELECT. We give the proof in Appendix B for completeness.<sup>3</sup>

## 5 EXPERIMENTS

We conduct experiments on three graph data sets that were studied by Wang et al. (2013) and a version of the Enron e-mail data by Priebe et al..

<sup>2</sup>This is similar to a bounded RKHS norm with kernel  $C_0$  in Srinivas et al. (2012).

<sup>3</sup>An earlier version of this paper follows on to discuss bounds on vanilla GP-SOPT. These proofs used strategies from Contal et al. (2014) which were found to be incorrect. Therefore, they have been removed in the current version of the paper.

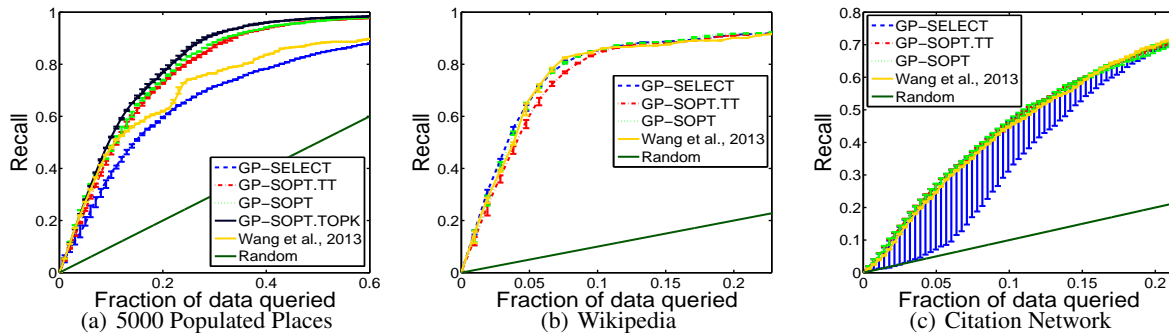


Figure 3: Recall vs. fraction of data queried

### 5.1 Three Graph Datasets of Wang et al. (2013)

We briefly summarize the datasets below.

**5000 Populated Places.** The nodes of this graph are 5000 concepts in the DBpedia<sup>4</sup> ontology marked as populated places. Each place is supported by a Wikipedia page, and an undirected edge is created between two places if either one of their two Wikipedia pages links to the other. There can be multiple edges between two places. The DBpedia ontology divides populated places into five categories: administrative regions, countries, cities, towns and villages. The 725 administrative regions are selected as our target class while all the others are considered to be in null class.

**Citation Network.** This dataset consists of 14,117 papers in top Computer Science venues available on citeseer. The graph is created by adding an undirected edge between two papers if either one cites the other. The 1844 NIPS papers are chosen as our target class.

**Wikipedia Pages on Programming Languages.** A total of 5,271 Wikipedia pages related to programming languages are the nodes of this graph, and an undirected edge exists between two pages if they are linked together. Wang et al. (2013) performed topic modeling and chose the 202 pages related to objective oriented programming as our target class.

As demonstrated by Wang et al. (2013), the three graphs and their target label distributions exhibit qualitative differences and thus serve as good benchmarks. The citation network has many small components and target nodes appear in many of them, while the Wikipedia graph has large hubs and most target nodes reside in one of them. The graph of populated places lies in between these two extremes, with components of various sizes containing target nodes.

On all of the three data sets we compare two of the proposed methods: GP-SOPT.TT and GP-SOPT against GP-SELECT (GP-UCB without replacement) and the active

search algorithm (AS-on-Graph) by Wang et al. (2013). We only evaluate GP-SOPT.TOPK on the 5000 populated places data due to its heavy computation. For each dataset we perform 5 independent runs, each with a randomly chosen target node as the warm start seed. For the proposed methods and GP-SELECT, the main tuning parameters are the exploration-exploitation trade-off parameter  $\alpha_t$  and the observation noise variance  $\sigma^2$ . For GP-SOPT.TT and GP-SOPT.TOPK there is additionally the thresholding parameter  $k$ . We consider the following values for them. Populated Places:  $\alpha_t \in \{4, 2, 1, 0.1, 0.01, 0.001\}$ ,  $\sigma^2 \in \{1, 0.5, 0.25, 0.1\}$  and  $k \in \{200, 400, 800\}$ . Wikipedia:  $\alpha_t \in \{0.1, 0.01, 0.001\}$ ,  $\sigma^2 \in \{1, 0.5, 0.25, 0.1\}$  and  $k \in \{200, 400, 800\}$ . Citation Network:  $\alpha_t \in \{1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}\}$ ,  $\sigma^2 \in \{1, 0.5, 0.25, 0.1\}$  and  $k \in \{400, 800, 1600\}$ . Although in theory  $\alpha_t$  should be iteration-dependent, we find that a fixed value often performs well in practice. On all data sets we set the kernel regularization parameter  $\omega_0 = 0.01$ . Wang et al. (2013) algorithm has several parameters, and we only tune the exploration-exploitation trade-off parameter  $\alpha$ . It is set to 0.1 on Populated Places and Citation Network, and 0.0001 on Wikipedia, which are the best performing values. Other parameters are set based on Wang et al. (2013).

Results are in Figure 3, where we plot the recall, i.e., the fraction of targets found by the algorithms, versus the fraction of the whole data set queried. More specifically, for each algorithm we obtain its mean recall curve over the top 15% (except for Wang et al. (2013)) parameter combinations in each experiment, as judged by the area under the recall curve. We then plot the median, maximum and minimum over the five runs in Figure 3.

The three proposed methods clearly outperform Wang et al. (2013) and GP-SELECT on Populated Places, while all methods perform equally well on Wikipedia. We think this has to do with the underlying graph structure and target distribution. As mentioned before, target nodes in the Populated Places graph are spread over sub-graphs of various

<sup>4</sup>www.dbpedia.org

sizes, and therefore exploration strategies do make a difference. We observe that the proposed methods tend to select high-degree nodes in the first few iterations, thereby gaining much information, while GP-SELECT initially selects low-degree nodes. In contrast, most target nodes in the Wikipedia graph reside in one large component, and therefore less exploration is needed. In fact, the best values for  $\alpha_t$  are very small, suggesting that an exploitation-only strategy is good enough for this data. On Citation Network, most methods perform well except that GP-SELECT performs quite poorly in one run. This may again indicate GP-SELECT is less robust against low-degree nodes.

## 5.2 Enron E-mails

We experimented on the Enron e-mail data set<sup>5</sup> with topics assigned by Priebe et al. based on the annotations by Berry and Browne. We further processed the dataset into a format suitable for active search experiments as detailed below. Each e-mail  $i$  is represented by a unique Unix time stamp  $t_i$ , a unique sender index and the set of receiver (excluding self-copying) indices, which are collectively denoted as  $U_i$ . Between e-mails  $i$  and  $j$ , we created an edge with the following weight:

$$A_{ij} := \exp(-(t_i - t_j)^2 / \tau^2) \cdot |U_i \cap U_j| / \sqrt{|U_i||U_j|},$$

where  $\tau = 12$  weeks in seconds and  $|U_i|$  denotes the size of  $U_i$ . We thus measure pairwise similarity among e-mails by the product of nearness in time and degree of overlap between users involved. The resulting e-mail graph has 20,112 nodes, and we chose the subset of 803 e-mails that are assigned topic 16 in LDC topics<sup>5</sup>, which is related to the downfall of Enron, to be the target class in this experiment.

Due to the size of the dataset, we only compared three methods: GP-SOPT.TT, GP-SELECT and Wang et al. (2013) in three independent runs each initialized with a target node chosen uniformly at random. We also limited the tuning parameters to be the following fixed values across the three runs:  $(k, \alpha, \sigma^2, \omega_0) = (800, 0.001, 0.05, 0.01)$  for GP-SOPT.TT,  $(\alpha, \sigma^2, \omega_0) = (0.01, 0.05, 0.01)$  for GP-SELECT, and  $\alpha = 0.001$  for Wang et al. (2013). These values were chosen based on a coarse parameter search to be indicative of the performance of each method on this data set. Results are in Figure 4, which shows GP-SOPT.TT is more stable across initial seeds than the other methods, and outperforms Wang et al. (2013) significantly at early iterations.

## 6 CONCLUSION AND DISCUSSIONS

In this paper, we discuss active search on a graph with known structure. Each node bears a reward, which is unknown at first but can be noisily observed upon query. An

<sup>5</sup>Available at <http://cis.jhu.edu/~parky/Enron/execs.email.linesnum.ldctopic>

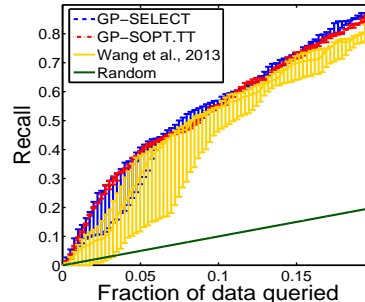


Figure 4: Enron: recall vs. fraction of data queried

active search algorithm aims to accumulate as large a sum of rewards from the queried nodes as possible under limited budgets. We assume that the node rewards vary smoothly along the graph.

Popular Bayesian UCB-style algorithms (Srinivas et al., 2012; Vanchinathan et al., 2013; Valko et al., 2014) use the marginal standard deviation as their exploration criterion, leading to the undesirable tendency of selecting peripheral nodes on a graph. Instead, we consider  $\Sigma$ -optimality on graphs, which can more efficiently reduce the variance of the reward function estimate by sampling cluster centers. We show the advantage of our method in experiments with real graphs and provide a theoretical guarantee on the cumulative regret.

One interesting future direction is deriving tighter regret bounds for the proposed methods that match their empirical performances. We imagine it may be possible to bound the regret directly by the difference in  $\Sigma$ -optimality (Bayes survey risks,  $\mathcal{R}^\Sigma$ ), which may have better properties than differential information gain,  $\gamma_T$  on graphs.

An equally interesting question is the selection of graph kernels. Our discussions and experiments mainly consider Gaussian random fields with unnormalized Laplacian, which is a very popular kernel choice. It is worthwhile to explore active search with other graph kernels, such as the ones discussed in Smola and Kondor (2003).

### Acknowledgement

This work was funded in part by DARPA grant FA87501220324.

## A Predictive Covariance Matrix

**Lemma 3.** For augmented graph Laplacian, the posterior covariance matrix,  $C_t(v, v') \geq 0, \forall v, v'$ .

*Proof.* Let  $h_k = \sum_{\tau=1}^t e_{v_\tau}(v_k)$  to be the count of queries on node  $k$ ; further define its diagonal matrix,  $\mathbf{H} =$

$\text{diag}(h_1, \dots, h_n)$ . We rewrite (5) as,

$$(\mathbf{C}_t)^{-1} = (\mathbf{C}_0)^{-1} + \sigma_n^{-2} \mathbf{H} = \mathbf{D} - \mathbf{A} + \omega_0 \mathbf{I} + \sigma_n^{-2} \mathbf{H}$$

Define  $\mathbf{D}_t = \mathbf{D} + \omega_0 \mathbf{I} + \sigma_n^{-2} \mathbf{H}$ , we have

$$\mathbf{C}_t = (\mathbf{D}_t - \mathbf{A})^{-1} = \mathbf{D}_t^{-\frac{1}{2}} \left( \sum_{k=0}^{\infty} \left( \mathbf{D}_t^{-\frac{1}{2}} \mathbf{A} \mathbf{D}_t^{-\frac{1}{2}} \right)^k \right) \mathbf{D}_t^{-\frac{1}{2}},$$

where the right hand side is always nonnegative.

The convergence of  $\|\mathbf{D}_t^{-\frac{1}{2}} \mathbf{A} \mathbf{D}_t^{-\frac{1}{2}}\|_2 < 1$  is as follows.

Define the components for the posterior as  $\mathbf{D}_t = \text{diag}(d_1^{(t)}, \dots, d_n^{(t)})$  with  $d^{(t)} = \sum_{i=1}^n d_i^{(t)}$ . Also, define for the prior model  $\mathbf{D} = \text{diag}(d_1^{(0)}, \dots, d_n^{(0)})$  with  $d^{(0)} = \sum_{i=1}^n d_i^{(0)}$ .

The following holds for any  $\mathbf{v} \in \mathbb{R}^n$ ,

$$\begin{aligned} \mathbf{v}^\top \mathbf{D}_t^{-\frac{1}{2}} \mathbf{A} \mathbf{D}_t^{-\frac{1}{2}} \mathbf{v} &= \sum_{ij} \frac{v_i v_j a_{ij}}{\sqrt{d_i^{(t)}} \sqrt{d_j^{(t)}}} \\ &\leq \sqrt{\left( \sum_{ij} \frac{v_i^2 a_{ij}}{d_i^{(t)}} \right) \left( \sum_{ij} \frac{v_j^2 a_{ij}}{d_j^{(t)}} \right)} = \sum_i v_i^2 \frac{d_i}{d_i^{(t)}} \leq \|\mathbf{v}\|_2^2. \end{aligned}$$

Further, both equalities cannot hold simultaneously, because for the first equality to hold, it is required that  $\frac{v_i^2 a_{ij}}{d_i^{(t)}} \propto \frac{v_j^2 a_{ij}}{d_j^{(t)}}$ , i.e.,  $v_j^2 \propto d_j^{(t)}$ ,  $\forall j$  in the same connected component, which then dictates that,

$$\sum_i v_i^2 \frac{d_i}{d_i^{(t)}} = \sum_i \left( \frac{d_i^{(0)}}{d^{(0)}} \|\mathbf{v}\|_2^2 \right) \frac{d_i}{d_i^{(t)}} = \frac{d^{(0)}}{d^{(t)}} \|\mathbf{v}\|_2^2 < \|\mathbf{v}\|_2^2. \quad \square$$

**Lemma 4.** *The diagonal elements in  $\mathbf{C}_t$  is always no smaller than the off-diagonal elements, i.e.,  $\sigma_t(v)^2 = C_t(v, v) \geq C_t(v, v'), \forall v, v'$ .*

*Proof.* Without loss of generality, let  $v$  be the last index of  $\mathbf{C}_t = (\tilde{\mathcal{L}}_0 + \sigma_n^{-2} \mathbf{H})^{-1}$ . For simplicity, let  $\tilde{\mathcal{L}}_t = \tilde{\mathcal{L}}_0 + \sigma_n^{-2} \mathbf{H}$  and it has the following matrix partition,

$$\tilde{\mathcal{L}}_t = \begin{pmatrix} \tilde{\mathcal{L}}_{\bar{v}\bar{v}} & \tilde{\ell}_{\bar{v}v} \\ \tilde{\ell}_{\bar{v}v}^\top & \tilde{\ell}_{vv} \end{pmatrix},$$

where  $\bar{v}$  is the complement of  $v$ . From Woodbury matrix inversion lemma, we have

$$\mathbf{C}_t = \tilde{\mathcal{L}}_t^{-1} = \begin{pmatrix} \mathbf{M} & -\frac{1}{m} \tilde{\mathcal{L}}_{\bar{v}\bar{v}}^{-1} \tilde{\ell}_{\bar{v}v} \\ -\frac{1}{m} \tilde{\ell}_{\bar{v}v}^\top \tilde{\mathcal{L}}_{\bar{v}\bar{v}}^{-1} & \frac{1}{m} \end{pmatrix}, \quad (16)$$

where  $m = \tilde{\ell}_{vv} - \tilde{\ell}_{\bar{v}v}^\top \tilde{\mathcal{L}}_{\bar{v}\bar{v}}^{-1} \tilde{\ell}_{\bar{v}v}$  and  $\mathbf{M} = \tilde{\mathcal{L}}_{\bar{v}\bar{v}}^{-1} + \frac{1}{m} \tilde{\mathcal{L}}_{\bar{v}\bar{v}}^{-1} \tilde{\ell}_{\bar{v}v} \tilde{\ell}_{\bar{v}v}^\top \tilde{\mathcal{L}}_{\bar{v}\bar{v}}^{-1}$ . To show that  $C_t(v, v) \geq C_t(v, v')$ , we need to verify that  $(-\tilde{\mathcal{L}}_{\bar{v}\bar{v}}^{-1} \tilde{\ell}_{\bar{v}v})_{v'} \leq 1$ .

In fact, since  $\tilde{\mathcal{L}}_t$  is diagonally dominant, we have  $\tilde{\mathcal{L}}_t \mathbf{1}_n \geq 0$ . Take its first  $n-1$  rows to get  $\tilde{\mathcal{L}}_{\bar{v}\bar{v}} \cdot \mathbf{1}_{n-1} + \tilde{\ell}_{\bar{v}v} \geq 0$ . Notice  $\tilde{\mathcal{L}}_{\bar{v}\bar{v}}$  is also a valid augmented graph Laplacian. By Lemma 3, we could left multiply the element-wise nonnegative matrix  $\tilde{\mathcal{L}}_{\bar{v}\bar{v}}^{-1}$  to both sides to obtain,  $\mathbf{1}_{n-1} + \tilde{\mathcal{L}}_{\bar{v}\bar{v}}^{-1} \tilde{\ell}_{\bar{v}v} \geq 0$ , which completes our proof for any  $v' \in \bar{v}$ .  $\square$

## B Active Search Regret Bound

We start by stating the following result.

**Theorem 5** (Theorem 6, Srinivas et al. (2012)). *Let  $\delta \in (0, 1)$ . Assume the observation noises are uniformly bounded by  $\sigma_n$  and  $f$  has RKHS norm  $B$  with kernel  $C_0$ , which is equivalent to  $\mathbf{f}^\top \tilde{\mathcal{L}}_0 \mathbf{f} \leq B^2$ . Define  $\alpha_t = \sqrt{2B^2 + 300\gamma_t \log(t/\delta)^3}$ , then*

$$\Pr(\forall t, \forall v \in V, |\mu_t(v) - f(v)| \leq \alpha_{t+1} \sigma_t(v)) \geq 1 - \delta.$$

We use this result to bound our instantaneous regrets.

**Lemma 6.** *Conditioned on the high-probability event in Theorem 5, the following bound holds:*

$$\forall t, r_t := f(v_t^*) - f(v_t) \leq 2\alpha_t k \sigma_{t-1}(v_t),$$

where  $v_t^*$  is the node with the  $t$ -th globally largest function value and  $v_t$  is node selected at round  $t$ .

*Proof.* At round  $t$  there are two possible situations. If  $v_t^*$  was picked at some earlier round, the definition of  $v_t^*$  implies that there exists some  $t' < t$  such that  $v_{t'}^*$  has not been picked yet. According to our selection rule, the fact that  $s_t(v) \geq \sigma_t(v)$ , and Theorem 5, the following holds:

$$\begin{aligned} \mu_{t-1}(v_t) + \alpha_t s_{t-1}(v_t) &\geq \mu_{t-1}(v_{t'}^*) + \alpha_t s_{t-1}(v_{t'}^*) \\ &\geq \mu_{t-1}(v_{t'}^*) + \alpha_t \sigma_{t-1}(v_{t'}^*) \geq f(v_{t'}^*) \geq f(v_t^*). \end{aligned}$$

If  $v_t^*$  has not been picked yet, a similar argument gives

$$\mu_{t-1}(v_t) + \alpha_t s_{t-1}(v_t) \geq \mu_{t-1}(v_t^*) + \alpha_t s_{t-1}(v_t^*) \geq f(v_t^*).$$

Thus we always have

$$\begin{aligned} f(v_t^*) &\leq \mu_{t-1}(v_t) + \alpha_t s_{t-1}(v_t) \\ &\leq f(v_t) + \alpha_t \sigma_{t-1}(v - t) + \alpha_t s_{t-1}(v_t) \\ &\leq f(v_t) + 2\alpha_t k \sigma_{t-1}(v_t). \end{aligned}$$

**Lemma 7** (Lemma 5.4, Srinivas et al. (2012)). *Let  $\alpha_t$  be defined as in Theorem 5 and  $c_1$  be defined as in Theorem 2. Conditioned on the high-probability event of Theorem 5, the following holds:*

$$\forall T \geq 1, \sum_{t=1}^T r_t^2 \leq \alpha_T k^2 c_1 \mathcal{I}(\mathbf{y}_{\mathbf{v}_T}; f_{\mathbf{v}_T}) \leq \alpha_T k^2 c_1 \gamma_T.$$

Finally, the Cauchy-Schwarz inequality gives  $R_T \leq \sqrt{T \sum_{t=1}^T r_t^2} \leq k \sqrt{T c_1 \alpha_T \gamma_T}$ .



## References

- Peter Auer. Using confidence bounds for exploitation-exploration trade-offs. *The Journal of Machine Learning Research*, 3:397–422, 2003.
- Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *NIPS*, volume 14, pages 585–591, 2001.
- Michael W. Berry and Murray Browne. The 2001 annotated (by topic) Enron email data set. URL [http://cis.jhu.edu/~parky/Enron/Anno\\_Topic\\_exp\\_LDC.pdf](http://cis.jhu.edu/~parky/Enron/Anno_Topic_exp_LDC.pdf).
- Sébastien Bubeck, Gilles Stoltz, Csaba Szepesvári, and Rémi Munos. Online optimization in X-armed bandits. In *Advances in Neural Information Processing Systems*, pages 201–208, 2009.
- Emile Contal, Vianney Perchet, and Nicolas Vayatis. Gaussian process optimization with mutual information. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 253–261, 2014.
- Dennis D Cox and Susan John. Sdo: A statistical method for global optimization. *Multidisciplinary design optimization: state of the art*, pages 315–329, 1997.
- Varsha Dani, Thomas P Hayes, and Sham M Kakade. Stochastic linear optimization under bandit feedback. In *COLT*, pages 355–366, 2008.
- Roman Garnett, Yamuna Krishnamurthy, Xuehan Xiong, Jeff Schneider, and Richard Mann. Bayesian optimal active search and surveying. In *ICML*, 2012.
- Alkis Gotovos, Nathalie Casati, Gregory Hitz, and Andreas Krause. Active learning for level set estimation. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, pages 1344–1350. AAAI Press, 2013.
- Ming Ji and Jiawei Han. A variance minimization criterion to active learning on graphs. In *AISTAT*, 2012.
- Robert Kleinberg, Aleksandrs Slivkins, and Eli Upfal. Multi-armed bandits in metric spaces. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 681–690. ACM, 2008.
- Andreas Krause, Ajit Singh, and Carlos Guestrin. Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research (JMLR)*, 9:235–284, February 2008.
- Yifei Ma, Roman Garnett, and Jeff Schneider.  $\Sigma$ -optimality in active learning on Gaussian random fields. In *NIPS*, 2013.
- Carey E. Priebe, John M. Conroy, David J. Marchette, and Youngser Park. Scan statistics on Enron graphs. URL <http://cis.jhu.edu/~parky/Enron/enron.html>.
- Friedrich Pukelsheim. *Optimal design of experiments*, volume 50. siam, 1993.
- Carl Edward Rasmussen and Christopher KI Williams. *Gaussian processes for machine learning*, volume 1. MIT press Cambridge, MA, 2006.
- Herbert Robbins. Some aspects of the sequential design of experiments. In *Herbert Robbins Selected Papers*, pages 169–177. Springer, 1985.
- Burr Settles. Active learning literature survey. *University of Wisconsin, Madison*, 52(55-66):11, 2010.
- Alexander J. Smola and Risi Kondor. Kernels and regularization on graphs. In *COLT/Kernel*, pages 144–158, 2003.
- Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Information-theoretic regret bounds for gaussian process optimization in the bandit setting. *Information Theory, IEEE Transactions on*, 58(5):3250–3265, 2012.
- Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.
- Michal Valko, Rémi Munos, Branislav Kveton, and Tomáš Kocák. Spectral bandits for smooth graph functions. In *31th International Conference on Machine Learning*, 2014.
- Hastagiri P Vanchinathan, Andreas Marfurt, Charles-Antoine Robelin, Donald Kossman, and Andreas Krause. Adaptively selecting valuable diverse sets via Gaussian processes and submodularity. In *NIPS Workshop on Discrete and Combinatorial Problems in Machine Learning (DISCML) 2013: Theory and Applications*, 2013.
- Xuezhi Wang, Roman Garnett, and Jeff Schneider. Active search on graphs. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 731–738. ACM, 2013.
- Xiaojin Zhu, Zoubin Ghahramani, John Lafferty, et al. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML*, volume 3, pages 912–919, 2003a.
- Xiaojin Zhu, John Lafferty, and Zoubin Ghahramani. Combining active learning and semi-supervised learning using gaussian fields and harmonic functions. In *ICML 2003 workshop on the continuum from labeled to unlabeled data in machine learning and data mining*, pages 58–65, 2003b.

---

# Off-policy learning based on weighted importance sampling with linear computational complexity

---

A. Rupam Mahmood

Richard S. Sutton

Reinforcement Learning and Artificial Intelligence Laboratory

Department of Computing Science, University of Alberta, Edmonton, AB T6G 2E8 Canada

## Abstract

Importance sampling is an essential component of model-free off-policy learning algorithms. Weighted importance sampling (WIS) is generally considered superior to ordinary importance sampling but, when combined with function approximation, it has hitherto required computational complexity that is  $O(n^2)$  or more in the number of features. In this paper we introduce new off-policy learning algorithms that obtain the benefits of WIS with  $O(n)$  computational complexity. Our algorithms maintain for each component of the parameter vector a measure of the extent to which that component has been used in previous examples. This measure is used to determine component-wise step sizes, merging the ideas of stochastic gradient descent and sample averages. We present our main WIS-based algorithm first in an intuitive acausal form (the forward view) and then derive a causal algorithm using eligibility traces that is equivalent but more efficient (the backward view). In three small experiments, our algorithms performed significantly better than prior  $O(n)$  algorithms for off-policy policy evaluation. We also show that our adaptive step-size technique can also improve the performance of *on-policy* algorithms such as TD( $\lambda$ ) and true online TD( $\lambda$ ).

## 1 Weighted importance sampling for off-policy Monte Carlo estimation

In off-policy learning problems, an agent learns about a policy while its experience is generated by following a different policy. A Monte Carlo technique known as *importance sampling* (Kahn & Marshall 1953, Rubinstein 1981) is often used to resolve this mismatch in policies (Sutton & Barto 1998, Dann, Neumann & Peters 2014, Geist & Scherrer 2014). One of the most effective variants

of importance sampling is *weighted importance sampling* (WIS), which often gives much lower variance than the ordinary form of importance sampling and is generally preferred due to its superior empirical performance (Hesterberg 1988, Precup, Sutton & Singh 2000, Shelton 2001, Liu 2001, Robert & Casella 2004, Koller & Friedman 2009).

Parametric function approximation is widely used and viewed as essential for large-scale reinforcement learning applications. However, only recently has WIS been extended to this more general setting. Mahmood, van Hasselt and Sutton (2014) have recently developed WIS-LSTD( $\lambda$ ), which extends WIS from tabular off-policy learning to linear function approximation and eligibility traces. However, WIS-LSTD( $\lambda$ ) is a least-squares algorithm that involves a matrix inversion in its update, the computational complexity of which scales  $O(n^3)$  in the number of features. In large-scale applications,  $O(n)$  algorithms, such as stochastic gradient descent, temporal-difference (TD) learning, and its gradient-based variants, are often preferred.

WIS has not yet been extended to  $O(n)$  algorithms. Modern off-policy algorithms with  $O(n)$  computational complexity, such as GTD( $\lambda$ ) (Maei 2011), GQ( $\lambda$ ) (Maei & Sutton 2010), and true online GTD( $\lambda$ ) (van Hasselt, Mahmood & Sutton 2014), all use the ordinary variant of importance sampling and can suffer severely due to the problem of large variance (Defazio & Graepel 2014).

In this work, we take several steps to bridge the gap between the Monte Carlo estimator WIS and  $O(n)$  off-policy algorithms with function approximation. The key to this endeavor is to relate stochastic gradient descent (SGD) to simple Monte Carlo estimators such as the sample average. We realize that there is a missing link in that, when the function approximation setting reduces to the tabular setting, SGD does not reduce to the sample average estimator. This is not unique to off-policy learning, and the relationship is missing even in the on-policy setting. Our first key step is to develop a new SGD that bridges this gap in an on-policy supervised-learning setting. Using insights from this step, we subsequently develop  $O(n)$  off-policy algorithms based on WIS in a general reinforcement learning setting.

## 2 Sample averages and stochastic gradient descent

In this section, we investigate the relationship between the sample average estimator and SGD. We first show that SGD does not reduce to the sample average estimator in the fully-representable case also known as the *tabular* representation. Then we propose a modification to SGD and show that it achieves the sample average estimator when the feature representation is tabular.

The sample average is one of the simplest Monte Carlo estimators. In order to introduce it, consider that data arrives as a sequence of samples  $Y_k \in \mathbb{R}$  drawn from a fixed distribution. The goal of the learner is to estimate the expected value of the samples,  $v \doteq \mathbb{E}[Y_k]$ . The sample average estimator  $\hat{V}_{t+1}$  for data given up to time  $t$  can be defined and incrementally updated in the following way:

$$\hat{V}_{t+1} \doteq \frac{\sum_{k=1}^t Y_k}{t} = \hat{V}_t + \frac{1}{t} (Y_t - \hat{V}_t); \hat{V}_1 \doteq 0. \quad (1)$$

In the incremental update,  $\frac{1}{t}$  can be viewed as a form of step size, modulating the amount of change made to the current estimate, which decreases with time in this case. In the parametric function approximation case, we have to go beyond sample average and use stochastic approximation methods such as SGD.

To introduce SGD, we use a supervised-learning setting with linear function approximation. In this setting, data arrives as a sequence of input-output pairs  $(X_k, Y_k)$ , where  $X_k$  takes values from a finite set  $\mathcal{X}$  and  $Y_k \in \mathbb{R}$ . The learner observes a feature representation of the inputs, where each input is mapped to a feature vector  $\phi_k \doteq \phi(X_k) \in \mathbb{R}^n$ . The goal of the learner is to estimate the conditional expectation of  $Y_k$  for each unique input  $x \in \mathcal{X}$  as a linear function of the features:  $\theta^\top \phi(x) \approx v(x) \doteq \mathbb{E}[Y_k | X_k = x]$ . SGD incrementally updates the parameter vector  $\theta \in \mathbb{R}^n$  at each time step  $t$  in the following way:

$$\theta_{t+1} \doteq \theta_t + \alpha_t (Y_t - \theta_t^\top \phi_t) \phi_t, \quad (2)$$

where  $\alpha_t > 0$  is a scalar step-size parameter, which is often set to a small constant. The per-update time and memory complexity of SGD is  $O(n)$ .

Linear function approximation includes tabular representations as a special case. For example, if the feature vectors are  $|\mathcal{X}|$ -dimensional standard basis vectors, then each feature uniquely represents an input, and the feature representation becomes tabular.

We are interested in finding whether SGD degenerates to sample average when the linear function approximation setting reduces to the tabular setting. Both incremental updates are in a form where the previous estimate is incremented with a product of an error and a step size. In the

SGD update, the product also has the feature vector as a factor, but in the tabular setting it simply selects the input for which an update is made.

A major difference between SGD and sample average is the ability of SGD to track under non-stationarity through the use of a constant step size. Typically, the step size of SGD is set to a constant value or decreased with time, where the latter does not work well under non-stationarity but is similar to how sample average works. While we attempt to accommodate sample average estimation more closely within SGD, it is also desirable to retain the tracking ability of SGD.

SGD clearly cannot achieve sample average with a constant step size. On the other hand, if we set the step-size parameter in the SGD update as  $\alpha_t = \frac{1}{t}$ , the SGD update still does not subsume the sample average. This is because, in the SGD update (2), time  $t$  is the total number of samples seen so far, whereas, in the sample average update (1), it is the number of samples seen so far only for one specific input.

We take two important steps to bridge the gap between the sample average update and the SGD update. First, we extend the sample average estimator to incorporate tracking through recency weighting, where the amount of weight assigned to the recent samples is modulated by a scalar recency-weighting constant. This new *recency-weighted average* estimator subsumes sample average as a special case and hence unifies both tracking and sample averaging. Second, we propose a variant of SGD that reduces to recency-weighted average in the tabular setting and still uses only  $O(n)$  per-update memory and computation.

Our proposed recency-weighted average estimator can be derived by minimizing an empirical mean squared objective with recency weighting:

$$\tilde{V}_{t+1} \doteq \arg \min_v \frac{1}{t} \sum_{k=1}^t (1-\eta)^{t-k} (Y_k - v)^2; 0 \leq \eta < 1.$$

Here, the recency-weighting constant  $\eta$  exponentially weights the past observations down and thus gives more weight to the recent samples. When  $\eta = 0$ , all samples are weighted equally. The recency-weighted average can be defined and incrementally updated as follows:

$$\tilde{V}_{t+1} = \frac{\sum_{k=1}^t (1-\eta)^{t-k} Y_k}{\sum_{k=1}^t (1-\eta)^{t-k}} = \tilde{V}_t + \frac{1}{\tilde{U}_{t+1}} (Y_t - \tilde{V}_t), \quad (3)$$

$$\tilde{U}_{t+1} \doteq (1-\eta)\tilde{U}_t + 1; \quad \tilde{U}_1 \doteq 0, \quad \tilde{V}_1 \doteq 0. \quad (4)$$

It is easy to see that the recency-weighted average is an unbiased estimator of  $v$ . Moreover, when  $\eta = 0$ , it reduces to the sample average estimator.

Now, we propose a modified SGD in the supervised-learning setting that for tabular representation reduces to the recency-weighted average. The updates are as follows:

$$\mathbf{u}_{t+1} \doteq (\mathbf{1} - \eta \phi_t \circ \phi_t) \circ \mathbf{u}_t + \phi_t \circ \phi_t, \quad (5)$$

$$\alpha_{t+1} \doteq \mathbf{1} \oslash \mathbf{u}_{t+1}, \quad (6)$$

$$\theta_{t+1} \doteq \theta_t + \alpha_{t+1} \circ (Y_t - \theta_t^\top \phi_t) \phi_t, \quad (7)$$

where  $\eta \geq 0$  is the recency-weighting factor,  $\circ$  is component-wise vector multiplication,  $\oslash$  is component-wise vector division where a division by zero results in zero, and  $\mathbf{1} \in \mathbb{R}^n$  is a vector of all ones. Here,  $\alpha_{t+1} \in \mathbb{R}^n$  is a vector step-size parameter, set as the vector division of  $\mathbf{1}$  by  $\mathbf{u}_{t+1} \in \mathbb{R}^n$ , which parallels  $\tilde{U}$  of the recency-weighted average. We call  $\mathbf{u}$  the *usage vector*, as it can be seen as an estimate of how much each feature is ‘‘used’’ over time by the update. We call this algorithm the *usage-based SGD* (U-SGD). Replacing a division by zero with zero in the step-size vector amounts to having no updates for the corresponding component. This makes sense because a zero in any component of  $\mathbf{u}$  can occur only at the beginning when  $\mathbf{u}$  is initialized to zero and the corresponding feature has not been activated yet. Once a feature is nonzero, the corresponding component of  $\alpha$  becomes positive and, with sufficiently small  $\eta$ , it always remains so.

In the following theorem, we show that U-SGD reduces to recency-weighted average in the tabular setting and hence is a generalization of the sample average estimator as well.

**Theorem 1** (Backward consistency of U-SGD with sample average). *If the feature representation is tabular, the vectors  $\mathbf{u}$  and  $\theta$  are initially set to zero, and  $0 \leq \eta < 1$ , then U-SGD defined by (5)-(7) degenerates to the recency-weighted average estimator defined by (3) and (4), in the sense that each component of the parameter vector  $\theta_{t+1}$  of U-SGD becomes the recency-weighted average estimator of the corresponding input.*

(Proved in Appendix A.1).

### 3 WIS and off-policy SGD

In this section, we carry over weighted importance sampling (WIS) to off-policy SGD, drawing from the ideas developed in the previous section. We introduce two new off-policy SGD algorithms based on WIS. The first one subsumes WIS fully but does not lead to an  $O(n)$  implementation, whereas the other algorithm is more amenable to an efficient implementation.

First we introduce both the ordinary importance sampling (OIS) and WIS. Importance sampling is a technique for estimating an expectation under one distribution using samples drawn from a different distribution. OIS estimates the expectation by forming a special kind of sample average. Consider that samples  $Y_k \in \mathbb{R}$  are drawn from a sample distribution  $l$ , but the goal of the learner is to estimate the expectation  $v_g \doteq \mathbf{E}_g [Y_k]$  under a different distribution  $g$ . OIS

estimates  $v_g$  by scaling each sample  $Y_k$  by the importance-sampling ratio  $W_k \doteq \frac{g(Y_k)}{l(Y_k)}$  and forming a sample average estimate of the scaled samples:

$$\tilde{V}_{t+1} \doteq \frac{\sum_{k=1}^t W_k Y_k}{t} = \tilde{V}_t + \frac{1}{t} (W_t Y_t - \tilde{V}_t); \quad \tilde{V}_1 \doteq 0.$$

WIS, on the other hand, estimates  $v_g$  by forming a weighted average estimate of the original samples. Its definition and incremental update are as follows:

$$\hat{V}_{t+1} \doteq \frac{\sum_{k=1}^t W_k Y_k}{\sum_{k=1}^t W_k} = \hat{V}_t + \frac{1}{\hat{U}_{t+1}} W_t (Y_t - \hat{V}_t),$$

$$\hat{U}_{t+1} \doteq \hat{U}_t + W_t; \quad \hat{U}_1 \doteq 0, \quad \hat{V}_1 \doteq 0.$$

If there is no discrepancy between the sample and the target distribution, then  $W_k = 1, \forall k$ , and both OIS and WIS become equivalent to the sample average estimator.

We derive the *recency-weighted WIS* as a solution to a mean squared objective with recency weighting and additionally importance sampling:

$$\bar{V}_{t+1} \doteq \arg \min_v \frac{1}{t} \sum_{k=1}^t (1 - \eta)^{t-k} W_k (Y_k - v)^2; \quad 0 \leq \eta < 1,$$

$$= \frac{\sum_{k=1}^t (1 - \eta)^{t-k} W_k Y_k}{\sum_{k=1}^t (1 - \eta)^{t-k} W_k}.$$

It is easy to see that, when  $\eta = 0$ , the recency-weighted WIS estimator reduces to WIS. Recency-weighted WIS can be updated incrementally in the following way:

$$\bar{V}_{t+1} = \bar{V}_t + \frac{1}{\bar{U}_{t+1}} W_t (Y_t - \bar{V}_t); \quad \bar{V}_1 \doteq 0, \quad (8)$$

$$\bar{U}_{t+1} \doteq (1 - \eta) \bar{U}_t + W_t; \quad \bar{U}_1 \doteq 0. \quad (9)$$

Now we introduce two variants of SGD based on WIS in a more general off-policy reinforcement learning setting with linear function approximation. In this setting, a learning agent interacts with an environment by taking an action  $A_k \in \mathcal{A}$  in a state of the environment  $S_k \in \mathcal{S}$  on each time step  $k$ . Here  $\mathcal{A}$  and  $\mathcal{S}$  are considered finite. Upon taking an action, the agent receives a scalar reward  $R_{k+1}$  and transitions to state  $S_{k+1}$ . Instead of observing the states directly, the agent observes a feature representation of the states where each state is mapped to a feature vector  $\phi_k \doteq \phi(S_k) \in \mathbb{R}^n$ . In an off-policy policy-evaluation problem, the agent takes actions based on a fixed behavior policy  $b(\cdot | S_k)$ . The goal is to estimate  $v_\pi(s)$  (the expected sum of the future discounted reward when starting in  $s$  and following  $\pi$ ) as a linear function of the features:

$$\theta^\top \phi(s) \approx v_\pi(s) \doteq \mathbf{E} [G_k | S_k = s, A_i \sim \pi(\cdot | S_i), \forall i],$$

$$G_k \doteq \sum_{i=k}^{\infty} \gamma^i R_{i+1} \prod_{j=k+1}^i \gamma(S_j),$$

where  $\theta \in \mathbb{R}^n$  is the parameter vector to learn, and  $\gamma_k \doteq \gamma(S_k) \in [0, 1]$  denotes a state-dependent discounting. In a general value-function setting, such state-dependent discounting can be used to denote termination in a state  $s$  by setting  $\gamma(s) = 0$  (Sutton et al. 2011).

In order to learn  $\theta$  model-free from samples, we need to use an importance sampling technique since the behavior policy and the target policy can be different. Given a partial trajectory from time  $k$  to  $t + 1$ :  $S_k, A_k, R_{k+1}, \dots, S_{t+1}$ , the importance-sampling ratio  $\rho_k^{t+1}$  is defined as the likelihood ratio of this trajectory starting at  $S_k$  under the target policy and the behavior policy:

$$\begin{aligned} \rho_k^{t+1} &\doteq \frac{\prod_{i=k}^t p(S_{i+1}|S_i, A_i) \pi(A_i|S_i)}{\prod_{i=k}^t p(S_{i+1}|S_i, A_i) b(A_i|S_i)} = \prod_{i=k}^t \frac{\pi(A_i|S_i)}{b(A_i|S_i)} \\ &= \prod_{i=k}^t \rho_i; \quad \rho_i \doteq \rho_i^{i+1}, \end{aligned}$$

where  $p$  is the state-transition probability function. The value  $v_\pi(s)$  can be estimated using returns scaled by the corresponding importance-sampling ratios. Consider that data is available up to step  $t + 1$  and no termination or discounting occurs by that time. A return originating from state  $S_k$  can be approximated by using a full discounting at the final step:  $\gamma_{t+1} = 0$ . Then a flat truncated return can be defined as (Sutton et al. 2014):

$$G_k^{t+1} \doteq \sum_{i=k}^t R_{i+1}.$$

When the states are visible and the number of states are small, the value  $v_\pi(s)$  for each state  $s$  can be estimated using importance sampling such as OIS, WIS or recency-weighted WIS by setting  $Y_k \doteq G_k^{t+1}$  and  $W_k \doteq \rho_k^{t+1}$ .

Now, we propose the first off-policy SGD based on WIS, which we call *WIS-SGD-1*. With  $0 \leq k < t + 1$  and  $\theta_0^t \doteq \theta_0, \forall t$ , the following updates define WIS-SGD-1:

$$\mathbf{u}_{k+1}^{t+1} \doteq (\mathbf{1} - \eta \phi_k \circ \phi_k) \circ \mathbf{u}_k^{t+1} + \rho_k^{t+1} \phi_k \circ \phi_k, \quad (10)$$

$$\alpha_{k+1}^{t+1} \doteq \mathbf{1} \circ \alpha_{k+1}^{t+1}, \quad (11)$$

$$\theta_{k+1}^{t+1} \doteq \theta_k + \alpha_{k+1}^{t+1} \circ \rho_k^{t+1} (G_k^{t+1} - \phi_k^\top \theta_k^{t+1}) \phi_k. \quad (12)$$

Similar to U-SGD, WIS-SGD-1 maintains a vector step size through the update of a usage vector, which in this case also includes the importance-sampling ratios. Unlike U-SGD, the parameters of WIS-SGD-1 use two time indices. The time index in the subscript corresponds to the time step of the prediction, and the time index in the superscript stands for the data horizon. In the following, we show that WIS-SGD-1 reduces to recency-weighted WIS, and hence to WIS as well, in the tabular setting.

**Theorem 2** (Backward consistency of WIS-SGD-1 with WIS). *If the feature representation is tabular, the vectors*

$\mathbf{u}$  and  $\theta$  are initially set to zero, and  $0 \leq \eta < 1$ , then *WIS-SGD-1 defined by (10)-(12) degenerates to recency-weighted WIS defined by (8) and (9) with  $Y_k \doteq G_k^{t+1}$  and  $W_k \doteq \rho_k^{t+1}$ , in the sense that each component of the parameter vector  $\theta_{t+1}^{t+1}$  of WIS-SGD-1 becomes the recency-weighted WIS estimator of the corresponding input. (Proved in Appendix A.2.)*

Now we focus on whether and how WIS-SGD-1 can be implemented efficiently. The updates as defined above cannot be computed in  $O(n)$  per time step. An update for step  $k$  requires computing an importance-sampling ratio and a flat truncated return that are available only at  $t + 1 > k$ . It can be computed by looking ahead into the future from  $k$ , but then the update becomes acausal. It can alternatively be computed by waiting until time  $t + 1$  and iterating for each  $k$ . But then the update made at  $t + 1$  becomes expensive, scaling linearly with  $t$ , that is,  $O(tn)$ .

Such updates, where samples are available in future from the time step when the update is made, are often known as *forward-view* updates (Sutton & Barto 1998). Forward-view updates are typically expensive, but for some forward-view updates it is possible to derive causal and efficient updates, known as *backward-view* updates, that compute exactly the same estimate at each time step. Classically these equivalences were achieved for offline updating. Van Seijen and Sutton (2014) showed that such equivalences can also be achieved in the online case.

Converting a forward-view update into an efficient backward-view update depends on combining the extra data available at  $t + 1$  with the current estimate  $\theta_t^t$  in an efficient way to give the next estimate  $\theta_{t+1}^{t+1}$ . For linear recursive updates, it is tantamount to unrolling both  $\theta_{t+1}^{t+1}$  and  $\theta_t^t$  and expressing their difference in a form that can be computed efficiently. It is often not possible to achieve such efficient backward-view updates, and we believe WIS-SGD-1 is one such case.

To appreciate why an efficient backward-view update of WIS-SGD-1 is not plausible, consider the update of  $\theta_t^t$  unrolled back to the beginning of time:

$$\begin{aligned} \theta_t^t &= (\mathbf{I} - \rho_{t-1}^t (\alpha_{t-1}^t \circ \phi_{t-1}) \phi_{t-1}^\top) \theta_{t-1}^t + \rho_{t-1}^t G_{t-1}^t \phi_{t-1} \\ &= \prod_{k=0}^{t-1} (\mathbf{I} - \rho_k^t (\alpha_{k+1}^t \circ \phi_k) \phi_k^\top) \theta_0^t \\ &\quad + \sum_{k=0}^{t-1} \rho_k^t G_k^t \prod_{j=k+1}^{t-1} (\mathbf{I} - \rho_j^t (\alpha_{j+1}^t \circ \phi_j) \phi_j^\top) \phi_k. \end{aligned}$$

In order to obtain  $\theta_{t+1}^{t+1}$  by combining the new data  $\phi_t$  and  $R_{t+1}$  with  $\theta_t^t$ , it is evident that each of the  $\rho_k^t (\alpha_{k+1}^t \circ \phi_k) \phi_k^\top$  terms in the first product needs to be replaced by  $\rho_{k+1}^{t+1} (\alpha_{k+1}^{t+1} \circ \phi_k) \phi_k^\top$ , which is unlikely to be achieved in an inexpensive way. This problem does not appear in previous algorithms with online equivalence such as true online

TD( $\lambda$ ) (van Seijen & Sutton 2014) or true online GTD( $\lambda$ ) (van Hasselt, Mahmood & Sutton 2014), because the terms involved in the product of the unrolled update in those algorithms do not involve *forward-view* terms, that is, they contain  $\rho_k$  and  $\alpha_{k+1}$  in those products instead of  $\rho_k^{t+1}$  and  $\alpha_{k+1}^{t+1}$ . This specific problem with WIS-SGD-1 is due to the fact that the error of the update in (12) is multiplied by the forward-view terms  $\rho_k^{t+1}$  and  $\alpha_{k+1}^{t+1}$ .

The observation we made in the above leads us to develop a second off-policy SGD. In this algorithm, first we replace the forward-view term  $\alpha_{k+1}^{t+1}$  from the update of  $\theta$  with  $\alpha_{k+1}^{k+1}$ . Second, instead of multiplying the terms in the error  $G_k^{t+1} - \phi_k^\top \theta_k^{t+1}$  with the same forward-view term  $\rho_k^{t+1}$ , we multiply the first term  $G_k^{t+1}$  by  $\rho_k^{t+1}$  and the second term  $\phi_k^\top \theta_k^{t+1}$  by  $\rho_k$ . To account for this discrepancy, we add two more terms in the error, and the resultant error of the new update becomes  $\rho_k^{t+1} G_k^{t+1} - \rho_k^{t+1} \phi_k^\top \theta_{k-1}^{k-1} + \rho_k \phi_k^\top \theta_{k-1}^{k-1} - \rho_k \phi_k^\top \theta_k^{t+1}$ . Here, the first two terms are approximating the WIS-SGD-1 error  $\rho_k^{t+1} (G_k^{t+1} - \phi_k^\top \theta_k^{t+1})$ , whereas the last two terms are adding a bias. Although this new algorithm no longer reduces to WIS in the tabular setting, it is developed based on WIS and still retains the main ideas behind recency-weighted WIS. Hence, we call this algorithm *WIS-SGD-2*. The following updates define WIS-SGD-2, with  $0 \leq k < t + 1$ :

$$\mathbf{u}_{k+1}^{t+1} \doteq (\mathbf{1} - \eta \phi_k \circ \phi_k) \circ \mathbf{u}_k^{t+1} + \rho_k^{t+1} \phi_k \circ \phi_k, \quad (13)$$

$$\alpha_{k+1} \doteq \mathbf{1} \oslash \mathbf{u}_{k+1}^{k+1}, \quad (14)$$

$$\delta_k^{t+1} \doteq \rho_k^{t+1} G_k^{t+1} - \rho_k^{t+1} \phi_k^\top \theta_{k-1} + \rho_k \phi_k^\top \theta_{k-1} - \rho_k \phi_k^\top \theta_k^{t+1}, \quad (15)$$

$$\theta_{k+1}^{t+1} \doteq \theta_k^{t+1} + \alpha_{k+1} \circ \delta_k^{t+1} \phi_k. \quad (16)$$

Here,  $\theta_k \doteq \theta_k^k$ , and  $\theta_{-1} = \mathbf{0}$ . It can be easily verified that, in the on-policy case, WIS-SGD-2 degenerates to U-SGD and hence retains the backward consistency with the sample average estimator.

Although this algorithm has much more plausibility of having an efficient backward view due to the careful modifications, it is not yet immediately clear how such a backward-view update can be obtained. Van Hasselt, Mahmood and Sutton (2014) introduced an online equivalence technique from which both true online TD( $\lambda$ ) and true online GTD( $\lambda$ ) can be derived. Their technique requires the target in the error to have a specific recurrence relation. Unfortunately, that specific relation does not hold for the target in WIS-SGD-2. A new technique is needed in order to derive an efficient backward view for WIS-SGD-2.

## 4 A new online equivalence technique

In this section, we introduce a new technique for deriving efficient backward views from online forward-view updates. We show that this technique subsumes the existing

technique for online equivalences (van Hasselt, Mahmood & Sutton 2014). The following theorem describes the new online equivalence technique (Proved in Appendix A.3).

**Theorem 3** (Online equivalence technique). *Consider any forward view that updates toward an interim scalar target  $Y_k^t$  with*

$$\theta_{k+1}^{t+1} \doteq \mathbf{F}_k \theta_k^{t+1} + Y_k^{t+1} \mathbf{w}_k + \mathbf{x}_k, \quad 0 \leq k < t + 1,$$

where  $\theta_0^t \doteq \theta_0$  for some initial  $\theta_0$ , and both  $\mathbf{F}_k \in \mathbb{R}^{n \times n}$  and  $\mathbf{w}_k \in \mathbb{R}^n$  can be computed using data available at  $k$ . Assume that the temporal difference  $Y_k^{t+1} - Y_k^t$  at  $k$  is related to the temporal difference at  $k + 1$  as follows:

$$Y_k^{t+1} - Y_k^t = d_{k+1} (Y_{k+1}^{t+1} - Y_{k+1}^t) + b_t g_k \prod_{j=k+1}^{t-1} c_j, \quad 0 \leq k < t,$$

where  $b_k, c_k, d_k$  and  $g_k$  are scalars that can be computed using data available at time  $k$ . Then the final weight  $\theta_{t+1} \doteq \theta_{t+1}^{t+1}$  can be computed through the following backward-view updates, with  $\mathbf{e}_{-1} \doteq \mathbf{0}$ ,  $\mathbf{d}_0 \doteq \mathbf{0}$ , and  $t \geq 0$ :

$$\mathbf{e}_t \doteq \mathbf{w}_t + d_t \mathbf{F}_t \mathbf{e}_{t-1},$$

$$\theta_{t+1} \doteq \mathbf{F}_t \theta_t + (Y_t^{t+1} - Y_t^t) \mathbf{e}_t + Y_t^t \mathbf{w}_t + b_t \mathbf{F}_t \mathbf{d}_t + \mathbf{x}_t,$$

$$\mathbf{d}_{t+1} \doteq c_t \mathbf{F}_t \mathbf{d}_t + g_t \mathbf{e}_t.$$

This equivalence technique allows producing backward views that contain a *dutch trace*  $\mathbf{e}$  (van Hasselt et al. 2014) and an extra set of weights  $\mathbf{d}$  known as *provisional weights* (Sutton et al. 2014) at the same time. In previous works (Sutton et al. 2014, Mahmood et al. 2014), the provisional weights appeared only in offline updates. Due to this online equivalence technique, this is the first time the provisional weights have emerged in online updates.

In the following theorem, we show that the new equivalence technique is a generalization of the existing equivalence technique developed by van Hasselt et al. (2014). Hence, it readily follows that the existing algorithms with an online equivalence, such as true online TD( $\lambda$ ) and true online GTD( $\lambda$ ), can be derived using the new equivalence technique. The proof is given in Appendix A.4.

**Theorem 4** (Generality of the new equivalence technique). *The online equivalence technique by van Hasselt et al. (2014, Theorem 1) can be retrieved as a special case of the online equivalence technique given in Theorem 3.*

## 5 A new off-policy TD( $\lambda$ ) based on WIS

In this section, we develop a new off-policy algorithm that generalizes WIS-SGD-2 to partial termination and bootstrapping. Then we use the new online equivalence technique to derive an equivalent  $O(n)$  backward-view update. We use a state-dependent bootstrapping parameter  $\lambda_k \doteq \lambda(S_k) \in [0, 1]$  in developing the new algorithm.

First, we construct the target, and then we define a new update for the usage vector  $\mathbf{u}$  in this more general setting.

Based on the general off-policy forward view by Sutton et al. (2014), we combine truncated returns  $G_k^{t+1}$  and truncated corrected returns  $G_k^{t+1} + \phi_{t+1}^\top \theta_t$  scaled by corresponding weights due to discounting, bootstrapping and importance sampling to develop an overall return:

$$\begin{aligned} G_{k,t+1}^\rho &\doteq \rho_k C_k^t \left( (1 - \gamma_{t+1}) G_k^{t+1} + \gamma_{t+1} (G_k^{t+1} + \phi_{t+1}^\top \theta_t) \right) \\ &+ \sum_{i=k+1}^t \rho_k C_k^{i-1} \left( (1 - \gamma_i) G_k^i + \gamma_i (1 - \lambda_i) (G_k^i + \phi_i^\top \theta_{i-1}) \right) \\ &- \rho_k \left( C_k^t + \sum_{i=k+1}^t C_k^{i-1} (1 - \gamma_i \lambda_i) - 1 \right) \phi_k^\top \theta_{k-1}, \quad (17) \end{aligned}$$

where  $C_k^t \doteq \prod_{j=k+1}^t \gamma_j \lambda_j \rho_j$ ,  $\theta_k \doteq \theta_k^k$ ,  $0 \leq k < t + 1$  and  $\theta_{-1} = \mathbf{0}$ . It can be readily verified that, when no bootstrapping is used, that is,  $\lambda_k = 1, \forall k$  and discounting occurs only at the data horizon  $t + 1$ , that is,  $\gamma_0 = \gamma_1 = \dots = \gamma_t = 1$  and  $\gamma_{t+1} = 0$ , then  $G_{k,t+1}^\rho = \rho_k^{t+1} G_k^{t+1} - \rho_k^{t+1} \phi_k^\top \theta_{k-1} + \rho_k \phi_k^\top \theta_{k-1}$ . Hence  $G_{k,t+1}^\rho$  is a strict generalization of the WIS-SGD-2 target to the state-dependent discounting and bootstrapping.

The usage vector  $\mathbf{u}$  of the WIS-SGD algorithms rescales the components of the parameter updates in order to clamp down the updates proportionally when they become large due to large importance-sampling ratios. However, when bootstrapping is used, larger trajectories are given smaller weights, and hence their corresponding importance-sampling ratios will have less severe effect on the updates. For example, when full bootstrapping is used, that is,  $\lambda_k = 0, \forall k$ , the overall return becomes  $G_{k,t+1}^\rho = \rho_k (R_{k+1} + \gamma_{k+1} \phi_{k+1}^\top \phi_k)$ , with an importance-sampling ratio of a one-transition long trajectory. In such cases, updating  $\mathbf{u}$  with the importance-sampling ratio of the full trajectory  $\rho_k^{t+1}$  is unnecessary. Hence, the amount of importance weighting in  $\mathbf{u}$  at each step should be modulated by the amount of discounting and bootstrapping.

Based on the overall return in (17) and the idea of discounting and bootstrapping-aware update of  $\mathbf{u}$  discussed above, we propose a new off-policy TD algorithm based on WIS, which we call *WIS-TD*( $\lambda$ ). It consists of the following forward-view updates:

$$\tilde{\rho}_k^{t+1} \doteq \rho_k \sum_{i=k+1}^t C_k^{i-1} (1 - \gamma_i \lambda_i) + \rho_k C_k^t; \quad \tilde{\rho}_t^t \doteq 0, \quad (18)$$

$$\mathbf{u}_{k+1}^{t+1} \doteq (\mathbf{1} - \eta \phi_k \circ \phi_k) \circ \mathbf{u}_k^{t+1} + \tilde{\rho}_k^{t+1} \phi_k \circ \phi_k, \quad (19)$$

$$\alpha_{k+1} \doteq \mathbf{1} \circ \mathbf{u}_{k+1}^{k+1}, \quad (20)$$

$$\theta_{k+1}^{t+1} \doteq \theta_k^{t+1} + \alpha_{k+1} \circ \left( G_{k,t+1}^\rho - \rho_k \phi_k^\top \theta_k^{t+1} \right) \phi_k. \quad (21)$$

It can be easily verified that, when no bootstrapping is used, that is,  $\lambda_k = 1, \forall k$  and discounting occurs only at the data

horizon  $t + 1$ , that is,  $\gamma_0 = \gamma_1 = \dots = \gamma_t = 1$  and  $\gamma_{t+1} = 0$ , then  $\tilde{\rho}_k^{t+1} = \rho_k^{t+1}$ , and we already showed that the target of WIS-TD( $\lambda$ )  $G_{k,t+1}^\rho$  reduces to the WIS-SGD-2 target in this case. Hence, WIS-TD( $\lambda$ ) subsumes WIS-SGD-2, establishing a direct backward consistency to sample average.

In the following, we apply the new online equivalence technique to the above forward-view update to derive an  $O(n)$  backward-view update computing the same parameter vector  $\theta_t$  at each  $t$ . For that, first we derive an  $O(n)$  backward-view update for the step size that computes the same  $\alpha_t$  as in the above algorithm at each  $t$ .

**Theorem 5** (Backward view update for  $\alpha_t$  of WIS-TD( $\lambda$ )). *The step-size vector  $\alpha_t$  computed by the following backward-view update and the forward-view update defined by (18) – (20) are equal at each step  $t$ :*

$$\begin{aligned} \mathbf{u}_{t+1} &\doteq (\mathbf{1} - \eta \phi_t \circ \phi_t) \circ \mathbf{u}_t + \rho_t \phi_t \circ \phi_t \\ &+ (\rho_t - 1) \gamma_t \lambda_t (\mathbf{1} - \eta \phi_t \circ \phi_t) \circ \mathbf{v}_t, \quad (22) \end{aligned}$$

$$\mathbf{v}_{t+1} \doteq \gamma_t \lambda_t \rho_t (\mathbf{1} - \eta \phi_t \circ \phi_t) \circ \mathbf{v}_t + \rho_t \phi_t \circ \phi_t, \quad (23)$$

$$\alpha_{t+1} \doteq \mathbf{1} \circ \mathbf{u}_{t+1}. \quad (24)$$

(Proved in Appendix A.5.)

Now, we derive an  $O(n)$  backward-view update that computes the same  $\theta_t^t$  as the above forward view.

**Theorem 6** (Backward view update for  $\theta_t^t$  of WIS-TD( $\lambda$ )). *The parameter vector  $\theta_t$  computed by the following backward-view update and the parameter vector  $\theta_t^t$  computed by the forward-view update defined by (17) and (21) are equal at every time step  $t$ :*

$$\begin{aligned} \mathbf{e}_t &\doteq \rho_t \alpha_{t+1} \circ \phi_t \\ &+ \gamma_t \lambda_t \rho_t (\mathbf{e}_{t-1} - \rho_t (\alpha_{t+1} \circ \phi_t) \phi_t^\top \mathbf{e}_{t-1}), \quad (25) \end{aligned}$$

$$\begin{aligned} \theta_{t+1} &\doteq \theta_t + \alpha_{t+1} \circ \rho_t (\theta_{t-1}^\top \phi_t - \theta_t^\top \phi_t) \phi_t \\ &+ (R_{t+1} + \gamma_{t+1} \theta_t^\top \phi_{t+1} - \theta_t^\top \phi_t) \mathbf{e}_t \\ &+ (\rho_t - 1) \gamma_t \lambda_t (\mathbf{d}_t - \rho_t (\alpha_{t+1} \circ \phi_t) \phi_t^\top \mathbf{d}_t), \quad (26) \end{aligned}$$

$$\begin{aligned} \mathbf{d}_{t+1} &\doteq \gamma_t \lambda_t \rho_t (\mathbf{d}_t - \rho_t (\alpha_{t+1} \circ \phi_t) \phi_t^\top \mathbf{d}_t) \\ &+ (R_{t+1} + \theta_t^\top \phi_{t+1} - \theta_{t-1}^\top \phi_t) \mathbf{e}_t. \quad (27) \end{aligned}$$

(Proved in Appendix A.6.)

The overall backward view of WIS-TD( $\lambda$ ) is defined by (22) – (27), and its complete description is given in Appendix A.7. Note that, Theorem 6 does not depend on how  $\alpha_{t+1}$  is set. The per-update time and memory complexity of WIS-TD( $\lambda$ ) is  $O(n)$ . An auxiliary parameter vector might be included in WIS-TD( $\lambda$ ) by making use of the  $\mathbf{x}_k$  vector of the online equivalence technique as was done by van Hasselt et al. (2014), but we do not explore this possibility here.

A seemingly related algorithm is fLSTD-SA (Prashanth, Korda & Munos 2014), which is an on-policy stochastic approximation method derived based on the on-policy

LSTD algorithm (Bradtke & Barto 1996) by randomizing the transition samples. One might speculate whether an  $O(n)$  off-policy stochastic-approximation algorithm based on WIS can be derived from WIS-LSTD( $\lambda$ ) by applying the techniques used in fLSTD-SA. However, the application of fLSTD-SA in the off-policy case does not appear to achieve any form of WIS. Updating a vector step size based on the usage of the features and importance-sampling weights is a distinctive aspect of our new algorithm and is essential for obtaining the benefits of WIS, which is absent in existing stochastic-approximation methods.

## 6 Extending existing algorithms based on the new adaptive step size

The vector step-size adaptation based on the update of the usage vector  $\mathbf{u}$  is only loosely coupled with WIS-TD( $\lambda$ ) and can be freely combined with existing off-policy algorithms as well as the on-policy ones. When combined with the existing algorithms, this step-size adaptation is expected to yield benefits due to the rescaling it performs according to the magnitude of importance-sampling weights and the frequency of feature activation.

We propose two new off-policy algorithms: *WIS-GTD*( $\lambda$ ) and *WIS-TO-GTD*( $\lambda$ ), based on GTD( $\lambda$ ) (Maei 2011) and true online GTD( $\lambda$ ) (van Hasselt et al. 2014), respectively. In both algorithms, we propose replacing the scalar step size of the main parameter vector with the vector step size according to (22) – (24). The scalar step-size parameter of the auxiliary parameter vector of GTD( $\lambda$ ) and true online GTD( $\lambda$ ) could also be replaced with the vector step size with a different recency-weighting factor, but we leave it out here. The descriptions of these two algorithms are given in Appendix A.7.

We propose two new on-policy algorithms: *usage-based TD*( $\lambda$ ) (U-TD( $\lambda$ )) and *usage-based true online TD*( $\lambda$ ) (U-TO-TD( $\lambda$ )), by combining the vector-step-size adaptation with two existing on-policy algorithms: TD( $\lambda$ ) (Sutton & Barto 1998) and true online TD( $\lambda$ ) (van Seijen & Sutton 2014), respectively. There are interesting interrelationships between these on-policy and off-policy algorithms. For example, WIS-GTD( $\lambda$ ) becomes equivalent to U-TD( $\lambda$ ) in the on-policy case when the second step-size parameter  $\beta = 0$ . On the other hand, WIS-TD( $\lambda$ ) directly degenerates to U-TO-TD( $\lambda$ ) in the on-policy case, whereas WIS-TO-GTD( $\lambda$ ) reduces to U-TO-TD( $\lambda$ ) in the on-policy case with  $\beta = 0$ . We provide the description of U-TD( $\lambda$ ) and U-TO-TD( $\lambda$ ) in Appendix A.7.

## 7 Experimental results

In this section we evaluate the new algorithms using two sets of experiments with off-policy and on-policy policy-evaluation tasks, respectively. Source code for both the

off-policy and on-policy experiments are available online<sup>1</sup>. In the first set of experiments, we compared the new off-policy algorithms: WIS-TD( $\lambda$ ), WIS-GTD( $\lambda$ ) and WIS-TO-GTD( $\lambda$ ) with two existing  $O(n)$  algorithms: GTD( $\lambda$ ) and true online GTD( $\lambda$ ) (TO-GTD( $\lambda$ )), and with two least squares algorithms: LSTD-TO( $\lambda$ ), an off-policy algorithm proposed by Dann, Neumann and Peters (2014), and WIS-LSTD( $\lambda$ ), an ideal extension of WIS. For evaluation, we created three off-policy policy-evaluation tasks.

The first task was constructed based on a random-walk Markov chain where the states can be imagined to be laid out on a horizontal line. There were 11 non-terminal states and two terminal states: on the left and the right ends of the chain. From each non-terminal state, there were two actions available: `left`, leads to the state to the left, and `right`, leads to the state to the right. The initial state was always set to the state in the middle of the chain. The reward was sparse: 0 for all transitions except for the right-most transition to the terminal state, where it was +1. The behavior policy was uniformly random between the two actions and the target policy chose `right` with 0.99 probability. No discounting was used. The feature vectors were binary representations of state indices. For 11 non-terminal states, each feature vector was of length  $\lfloor \log_2(11) \rfloor + 1 = 4$ , and these vectors for the states from left to right were  $(0, 0, 0, 1)^\top$ ,  $(0, 0, 1, 0)^\top$ ,  $(0, 0, 1, 1)^\top$ ,  $\dots$ ,  $(1, 0, 1, 1)^\top$ . The features were all zero for the terminal states.

The second and the third tasks were constructed using randomly generated MDPs. We represent a randomly generated MDP as  $(N, m, b, \Gamma)$  where  $N$  and  $m$  stand for the number of states and actions, respectively, and  $b$  is a branching factor denoting the number of next states for a given state-action pair. Here,  $\Gamma \in \mathbb{R}^{N \times N}$  is a diagonal matrix where the entries are the state-dependent discounting  $\gamma(\cdot)$  for each state. We use such a state-dependent discounting to denote termination under the target policy while experience continues seamlessly under the behavior policy. For each state, the next  $b$  states were chosen from total  $N$  states randomly without replacement, and the transition probabilities were generated by partitioning the unit interval at  $b - 1$  cut points which were selected uniformly randomly from  $[0, 1]$ . The rewards for a transition from a state-action pair to the next state were selected uniformly randomly from  $[0, 1]$  and kept deterministic. The behavior policy probabilities for different actions in a particular state were set using uniform random numbers from  $[10^{-15}, 1 + 10^{-15}]$  and normalized to sum to one. The target policy is much less stochastic: one of the actions in a particular state is chosen to have probability 0.99 and the rest of the actions are equiprobable.

<sup>1</sup>Source code for off-policy experiments is available at <http://github.com/armahmood/wis-td-experiments> and for on-policy experiments at <http://github.com/armahmood/usage-td-experiments>.



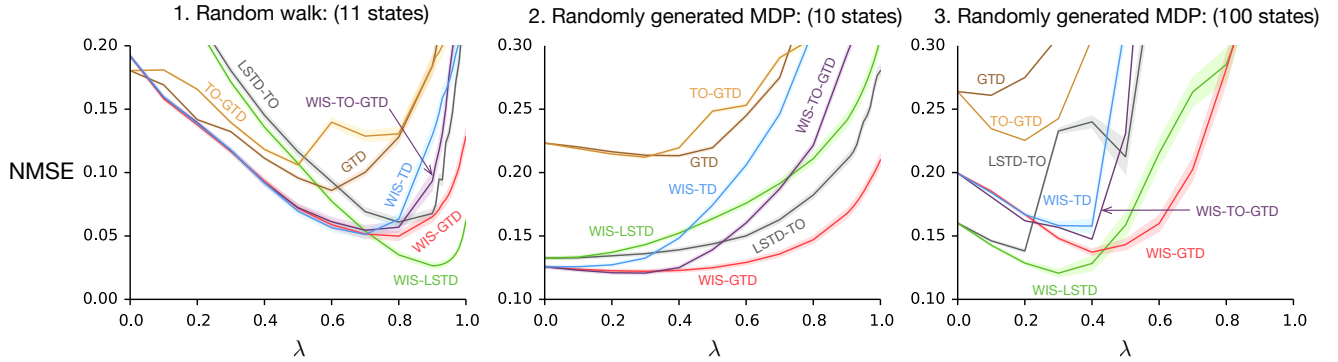


Figure 1: Empirical comparison of the new WIS-based  $O(n)$  algorithms with two existing  $O(n)$  algorithms and two LSTD algorithms on three off-policy policy-evaluation tasks. Performance is shown in the empirical normalized MSE (NMSE) measured by averaging over 50 independent runs and 100 episodes for the first task, 500 steps for the second, and 5000 steps for the third. The new WIS-based algorithms performed significantly better than both existing  $O(n)$  algorithms in all three off-policy tasks and competitively with one of the LSTD algorithms.

We constructed the second task by randomly generating an MDP with parameters  $(10, 3, 3, \Gamma)$ , where  $\gamma(\cdot) = 0$  for 2 randomly chosen states to denote termination under the target policy and  $\gamma(\cdot) = 0.99$  for the rest of the 8 states. For the third task, we randomly generated an MDP with parameters  $(100, 3, 10, \Gamma)$ , where  $\gamma(\cdot) = 0$  for 5 randomly chosen states and  $\gamma(\cdot) = 0.99$  for the rest. The feature vectors were binary representations of the indices of all states including those for which  $\gamma(\cdot) = 0$ . In these two tasks, the feature vectors were normalized to have unit length.

We tested all algorithms for different values of constant  $\lambda$ , from 0 to 0.9 in steps of 0.1 and from 0.9 to 1.0 in steps of 0.01. The first step-size parameter  $\alpha$  of  $\text{GTD}(\lambda)$  and  $\text{TO-GTD}(\lambda)$  was varied by powers of 10 with powers chosen from  $-3$  to 0 in steps of 0.25. The second step-size parameter  $\beta$  of both algorithms was varied among values  $[0, 0.001, 0.01, 0.1]$ . The initial value  $u_0$  of the components of the usage vector  $\mathbf{u}$  for  $\text{WIS-TD}(\lambda)$ ,  $\text{WIS-GTD}(\lambda)$  and  $\text{WIS-TO-GTD}(\lambda)$  was varied by powers of 10 with powers chosen from 0 to 3 in steps of 0.25. The recency-weighting factor  $\eta$  of the same algorithms was set as  $\eta = \mu/u_0$ , where  $\mu$  was varied among values  $[0, 0.001, 0.01, 0.1, 1]$ . The second step-size parameter  $\beta$  for  $\text{WIS-GTD}(\lambda)$  and  $\text{WIS-TO-GTD}(\lambda)$  was set to zero. The matrix to be inverted in  $\text{LSTD-TO}(\lambda)$  and  $\text{WIS-LSTD}(\lambda)$  was initialized to  $\epsilon \mathbf{I}$ , where  $\epsilon$  was varied by powers of 10 with powers chosen from  $-3$  to  $+3$  in steps of 0.2. The initial parameter vector  $\theta_0$  was set to  $\mathbf{0}$ .

Performance was measured as the empirical mean squared error (MSE) between the estimated values of the states and their true values under the target policy projected to the space spanned by the given features. The error was weighted according to the state-visitation distribution under the behavior policy. As the scale of this MSE measure can vary between these tasks, we normalized it by

the squared weighted L2 norm of the projected true value, which is equivalent to the MSE under  $\theta = \mathbf{0}$ . As a result, the initial normalized MSE (NMSE) for each algorithm was 1. For each run, we averaged this error over 100 episodes measured at the end of each episode for the first task, over 500 steps for the second task, and over 5000 steps for the third. We produced the final estimate by further averaging over 50 independent runs.

Figure 1 shows the empirical performance together with the standard error on the three off-policy policy-evaluation tasks with respect to different  $\lambda$  and optimized over all other parameters. In all three tasks, the new algorithms significantly outperformed both  $\text{GTD}(\lambda)$  and  $\text{TO-GTD}(\lambda)$  indicating the effectiveness of the adaptive vector step size in retaining the advantage of WIS. The new algorithms also performed competitively with  $\text{LSTD-TO}(\lambda)$  in all tasks. Among the new algorithms,  $\text{WIS-GTD}(\lambda)$  had superior performance with large values of  $\lambda$ .

We also studied the sensitivity of the new algorithms with respect to their parameters. Although these algorithms replace the scalar constant step-size parameter of their base learner with an adaptive vector step size based on feature usage, the estimate of the usage depends on two new parameters: the initial value  $u_0$  and the recency weighting constant  $\eta$ . The initial value  $u_0$  of the usage vector can be interpreted as the inverse of the initial step size, and its tuning can be as extensive as that of the scalar step-size parameter in other algorithms. On the other hand,  $\eta$  can be viewed as the desired final step size. As a result, their product  $\mu = u_0\eta$  is unit free and requires less rigorous tuning.

In our final set of experiments, we compared the new  $O(n)$  on-policy algorithms:  $\text{U-TD}(\lambda)$  and  $\text{U-TO-TD}(\lambda)$ , with two  $O(n)$  on-policy algorithms:  $\text{TD}(\lambda)$  with accumulating traces and true online  $\text{TD}(\lambda)$ , which we call  $\text{TO-TD}(\lambda)$ .

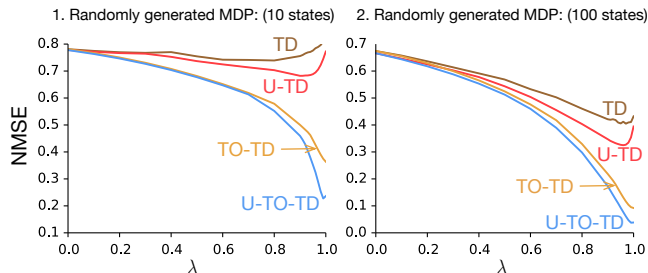


Figure 2: Empirical comparison of the new  $O(n)$  usage-based algorithms with two existing  $O(n)$  TD algorithms on on-policy policy-evaluation tasks. Performance is measured in empirical normalized MSE (NMSE).

We used randomly generated MDPs to produce two on-policy policy-evaluation tasks. As the TD algorithms here estimate state-value functions, it sufficed to construct Markov Reward Processes (MRPs), which we obtained by choosing the number of actions  $m$  to be 1 in both tasks. Our first task used an MDP with 10 states:  $(10, 1, 3, 0.99\mathbf{I})$  and the second task used an MDP with 100 states:  $(100, 1, 10, 0.99\mathbf{I})$ . The feature vectors were binary representations of the state indices as in the off-policy tasks and were normalized to have unit length.

For each task, the performance of each algorithm was measured for different parameter values. For  $\text{TD}(\lambda)$  and  $\text{TO-TD}(\lambda)$ , the scalar step-size parameter  $\alpha$  was varied by powers of 10 with powers chosen from  $-3$  to  $1$  in steps of  $0.25$ . For  $\text{U-TD}(\lambda)$  and  $\text{U-TO-TD}(\lambda)$ , the parameter  $u_0$  was varied by powers of 10 with powers chosen from  $-1$  to  $3$  in steps of  $0.25$ . The rest of the parameters for all four algorithms were varied using the same values as in the off-policy tasks. Performance was measured using NMSE as in the off-policy tasks. For each run, we averaged this error over 100 steps for the first task and 1000 steps for the second. The final estimate is produced again by averaging over 50 independent runs.

Figure 2 shows the performance on both tasks for different  $\lambda$  with the rest of the parameters optimized. Left plot corresponds to MDP  $(10, 1, 3, 0.99\mathbf{I})$  and the right plot corresponds to MDP  $(100, 1, 10, 0.99\mathbf{I})$ . On both tasks, the new algorithms performed significantly better than their base learning algorithms for higher values of  $\lambda$  and performed equally well for the smaller ones. The standard error in each case was smaller than the width of the curves shown. This set of experiments suggests that the step-size adaptation based on the usage of features can be useful in both off-policy and on-policy tasks.

## 8 Discussion and Conclusions

Weighted importance sampling (WIS), one of the most effective variants of importance sampling, has long been ne-

glected in off-policy learning with parametric function approximation. Recently introduced WIS-LSTD( $\lambda$ ) extends WIS to linear function approximation and eligibility traces but is  $O(n^3)$  in computational complexity in the number of features. In this paper, we took this endeavor one step further and carried over much of the benefit of WIS to  $O(n)$  off-policy algorithms. In the process, we developed modifications of stochastic gradient descent that are more closely related to sample averages. This endeavor also resulted in developing a new online equivalence technique for deriving causal efficient updates from acausal intuitive updates, which was deemed essential for achieving  $O(n)$  updates for our new algorithms. On three off-policy policy-evaluation experiments, the new algorithms outperformed the existing  $O(n)$  off-policy algorithms and performed competitively with LSTD-TO( $\lambda$ ).

An intriguing outcome of our work is an adaptive vector step size that is updated based on the *usage* of features, which has emerged naturally from our goal to incorporate the sample average within SGD. It is distinct from the existing adaptive step-size algorithms but not the only possible one that can achieve a Monte Carlo equivalence. An interesting direction for future work would be to explore the other possible variants. Although beyond the scope of this work, it is interesting to investigate how the insights from the new step-size adaptation idea can be incorporated in existing step-size adaptation algorithms such as Autostep (Mahmood et al. 2012), vSGD (Schaul et al. 2013) or Adam (Kingma et al. 2014). Convergence analysis of the new algorithms is another interesting direction for future work.

## Acknowledgements

The authors thank Hado van Hasselt, Joseph Modayil, Marlos C. Machado and Huizhen Yu for fruitful discussions that helped improve the quality of this work. This work was supported by grants from Alberta Innovates – Technology Futures, the National Science and Engineering Research Council of Canada, and the Alberta Innovates Centre for Machine Learning.

## References

Bradtke, S. J., Barto, A. G. (1996). Linear least-squares algorithms for temporal difference learning. *Machine Learning*, 22:33–57.

Dann, C., Neumann, G., Peters, J. (2014). Policy evaluation with temporal differences: a survey and comparison. *Journal of Machine Learning Research*, 15:809–883.

Defazio, A., Graepel, T. (2014). A comparison of learning algorithms on the Arcade Learning Environment. *arXiv preprint arXiv:1410.8620*.

- Geist, M., Scherrer, B. (2014). Off-policy learning with eligibility traces: A survey. *Journal of Machine Learning Research*, 15:289–333.
- Hesterberg, T. C. (1988). *Advances in Importance Sampling*, Ph.D. Dissertation, Statistics Department, Stanford University.
- Kahn, H., Marshall, A. W. (1953). Methods of reducing sample size in Monte Carlo computations. In *Journal of the Operations Research Society of America*, 1(5):263–278.
- Kingma, D. P., Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv:1412.6980*.
- Koller, D., Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.
- Liu, J. S. (2001). *Monte Carlo Strategies in Scientific Computing*. Berlin, Springer-Verlag.
- Maei, H. R., Sutton, R. S. (2010). GQ( $\lambda$ ): A general gradient algorithm for temporal-difference prediction learning with eligibility traces. In *Proceedings of the Third Conference on Artificial General Intelligence*, pp. 91–96. Atlantis Press.
- Maei, H. R. (2011). *Gradient Temporal-Difference Learning Algorithms*. PhD thesis, University of Alberta.
- Mahmood, A. R., Sutton, R. S., Degris, T., Pilarski, P. M. (2012). Tuning-free step-size adaptation. In *Proceedings of the 2012 IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 2121–2124.
- Mahmood, A. R., van Hasselt, H., Sutton, R. S. (2014). Weighted importance sampling for off-policy learning with linear function approximation. In *Advances in Neural Information Processing Systems 27*, Montreal, Canada.
- Prashanth, L. A., Korda, N., Munos, R. (2014). Fast LSTD using stochastic approximation: Finite time analysis and application to traffic control. In *Machine Learning and Knowledge Discovery in Databases* Springer, Lecture Notes in Computer Science, 8725:66–81.
- Precup, D., Sutton, R. S., Singh, S. (2000). Eligibility traces for off-policy policy evaluation. In *Proceedings of the 17th International Conference on Machine Learning*, pp. 759–766. Morgan Kaufmann.
- Robert, C. P., and Casella, G., (2004). *Monte Carlo Statistical Methods*, New York, Springer-Verlag.
- Rubinstein, R. Y. (1981). *Simulation and the Monte Carlo Method*, New York, Wiley.
- Schaul, T., Zhang, S., LeCun, Y. (2013). No more pesky learning rates. In *Proceedings of the 30th International Conference on Machine Learning*.
- Shelton, C. R. (2001). *Importance Sampling for Reinforcement Learning with Multiple Objectives*. PhD thesis, Massachusetts Institute of Technology.
- Sutton, R. S., Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. MIT Press.
- Sutton, R. S., Modayil, J., Delp, M., Degris, T., Pilarski, P. M., White, A., Precup, D. (2011). Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. In *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems*, pp. 761–768.
- Sutton, R. S., Mahmood, A. R., Precup, D., van Hasselt, H. (2014). A new Q( $\lambda$ ) with interim forward view and Monte Carlo equivalence. In *Proceedings of the 31st International Conference on Machine Learning*, Beijing, China.
- van Hasselt, H., Mahmood, A. R., Sutton, R. S. (2014). Off-policy TD( $\lambda$ ) with a true online equivalence. In *Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence*, Quebec City, Canada.
- van Seijen, H., & Sutton, R. S. (2014). True online TD( $\lambda$ ). In *Proceedings of the 31st International Conference on Machine Learning*. JMLR W&CP 32(1):692–700.

---

# Impact of Learning Strategies on the Quality of Bayesian Networks: An Empirical Evaluation

---

**Brandon Malone**  
Max Planck Institute  
for the Biology of Ageing

**Matti Järvisalo** and **Petri Myllymäki**  
HIIT, Department of Computer Science,  
University of Helsinki

## Abstract

We present results from an empirical evaluation of the impact of Bayesian network structure learning strategies on the learned structures. In particular, we investigate how learning algorithms with different optimality guarantees compare in terms of structural aspects and generalisability of the produced network structures. For example, in terms of generalization to unseen testing data, we show that local search algorithms often benefit from a tight constraint on the number of parents of variables in the networks, while exact approaches tend to benefit from looser parent restrictions. Overall, we find that learning strategies with weak optimality guarantees show good performance on synthetic datasets, but, compared to exact approaches, perform poorly on the more “real-world” datasets. The exact approaches, which guarantee to find globally optimal solutions, consistently generalize well to unseen testing data, motivating further work on increasing the robustness and scalability of such algorithmic approaches to Bayesian network structure learning.

## 1 INTRODUCTION

This work focuses on the well-known problem of learning the structure of a Bayesian network (BN) from data (Heckerman et al., 1995). The BN structure learning problem (BNSL) is to find a BN structure which optimizes a decomposable scoring function—typically, either a Bayesian posterior probability such as the Bayesian Dirichlet (BD) score or a penalized likelihood function. BNSL is NP-hard (Chickering, 1996), which poses challenges for developing algorithmic solutions for this important problem.

Methods proposed for solving BNSL on real-world datasets divide into approximate, local search methods, e.g., (Heckerman et al., 1995; Chickering, 2002; Teyssier and Koller,

2005; Tsamardinos et al., 2006), which do not offer quality guarantees (in terms of how well a given decomposable scoring function is optimized), and exact algorithms (Ott and Miyano, 2003; Koivisto and Sood, 2004; Silander and Myllymäki, 2006; Parviainen and Koivisto, 2009; de Campos and Ji, 2011; Yuan and Malone, 2013; Bartlett and Cussens, 2013) which produce guaranteed optimal solutions with respect to the scoring function. Often, the local search methods are computationally efficient, while the exact algorithms can require an exponential amount of time (and in cases, even memory).

The score of a BN structure is ideally a reflection of how well it models a training dataset. The general assumption has been that networks which model the training data well should also accurately reflect new data. However, it is well-known that a model can describe a training set very well, yet generalize poorly to new data (Mitchell, 1997). Thus, there is no guarantee that a network which optimizes a score for a training set will generalize well to new data.

There is currently no clear empirical evidence (for or against) on whether the increased computational efforts required by exact approaches to BNSL are justifiable in terms of performance of learned BNs on unseen testing data. All in all, the relationship between the chosen learning algorithm and the quality of the learned BN structures in terms of generalisability is not well understood. Furthermore, the choice of the learning algorithm used can affect structural properties of the learned networks in other ways. For example, for many of the exact algorithmic solutions to be applicable, in practice structural restrictions must be placed on the classes of BN structures considered during search. A commonly-applied restriction is a hard constraint on the number of parents allowed for each vertex in the network structures (Friedman et al., 1999). Since most scores incorporate a complexity penalty as a “soft constraint” favoring sparser networks, this is a practically-motivated restriction, as computing the scoring function for an arbitrary number of parents is in general infeasible (though pruning rules (de Campos and Ji, 2011) may in cases permit computing all necessary scores for datasets with few records).

However, the influence of such choices combined with the choice of the learning algorithm used—we refer to the combination of the two as a *learning strategy*—has not received much attention.

The aim of this work is to establish a more in-depth understanding of the aforementioned, currently not well-understood aspects of BNSL via an extensive empirical evaluation. Our aim is to shed light on the relationship of different learning strategies, based on four popular score-based structure learning algorithms, and the unknown discrepancy between training set scores and generalization. In particular, we address the following research questions for different fixed learning algorithms and training sets.

- Q1** How similar are structures found using different learning strategies?
- Q2** How do hard constraints on the number of parents in learned structures affect their generalization?
- Q3** How does the amount of training data affect the generalization of learned structures?
- Q4** Which learning strategies result in networks with the best generalization?

Our main contributions, based on a rigorous experimental setup, are the following. For Q1, we show that the different learning strategies tend to learn dissimilar network structures. With respect to Q2, we show that for small datasets, hard constraints limiting the maximum number of parents to 2 improves generalization on a few datasets for local search algorithms; however, optimal algorithms usually benefit from a higher limit. We answer Q3 by using increasingly large subsets of available training data. Regardless of the algorithms’ guarantees, more training data results in more accurate predictions on testing data. Finally, we address Q4 by considering all of the data collected during the evaluation. For some datasets, simple strategies such as the tractable Chow-Liu algorithm can provide good generalization. However, the simple strategies fail to generalize well on other datasets. Predictive likelihood results show that the optimal algorithms consistently generalize well.

While some of the observations made in this work, based on concrete empirical data, may appear to the reader as common knowledge, we note that to our best knowledge this is the first work which studies all of the question listed above thoroughly via empirical means. A previous related study is (Acid et al., 2004), which focuses on a case study of the choice of BNSL learning strategies for data from an emergency medical service. That study focused on Q1 for that particular domain and, orthogonally to the present work, the impact of the choice of scoring functions on the learning results. Furthermore, focusing on causal Bayesian networks, an evaluation of structural distance measures was presented in (de Jongh and Druzdzal, 2009). (Ueno, 2010;

Ueno, 2011) studied the effect of the equivalent sample size of BD on learned structures but did not consider predictive likelihood.

## 2 BACKGROUND

A Bayesian network (BN) (Pearl, 1988) is a compact representation of a joint probability distribution over random variables  $\mathbf{X} = \{X_1, \dots, X_n\}$ . It consists of a directed acyclic graph (DAG)  $G$ , in which each vertex in the graph corresponds to one of the random variables, and conditional probability distributions  $P(X_i|PA_i)$ , where  $PA_i$  is the set of parents of  $X_i$  in  $G$ . The (log) joint probability distribution over all of the variables is

$$\log P(X_1, \dots, X_n|G) = \sum_i^n \log P(X_i|PA_i).$$

Given a BN  $\mathcal{N}$  and a dataset  $d = \{d_1, \dots, d_N\}$ , where each  $d_r$  (record) is independent of the others, and a complete instantiation of  $\mathbf{X}$ , the likelihood of  $d$  given  $\mathcal{N}$  is

$$\log P(d|\mathcal{N}) = \sum_r^N \log P(d_r|\mathcal{N}) = \sum_r^N \sum_i^n \log P(X_i^r|PA_i^r), \quad (1)$$

where  $X_i^r$  and  $PA_i^r$  are the instantiations of  $X_i$  and its parents in record  $d_r$ , respectively. In total, Equation 1 consists of  $N \cdot n$  terms, each of which corresponds to the likelihood of one variable in one record given  $\mathcal{N}$ . We will refer to this as the *predictive likelihood* of  $\mathcal{N}$  on  $d$  and use the notation  $\ell^d$  when  $\mathcal{N}$  is clear from context.

Given a training dataset  $d_t$  and a scoring function  $s$ , the *Bayesian network structure learning problem* (BNSL) is to find a BN  $\mathcal{N}^* \in \arg \max_{\mathcal{N}} s(d_t, \mathcal{N})$ , i.e., a Bayesian network structure with the best score in terms of the scoring function  $s$ . The scoring function  $s$  is usually a Bayesian posterior probability or penalized likelihood function which measures how well  $\mathcal{N}$  “fits”  $d_t$ . In practice,  $s$  is *decomposable* (Heckerman et al., 1995), i.e.,  $s(d_t, \mathcal{N}) = \sum_i s(d_t, X_i, PA_i)$ . The  $s(d_t, X_i, PA_i)$  terms are often called *local scores*.

## 3 LEARNING ALGORITHMS

Our primary goal is to compare the impact of the guarantees of structure learning algorithms to the generalization of learned networks to unseen testing data. For this, we use four popular score-based learning algorithms with a range of optimality guarantees. In the following discussion, optimality guarantees refer to behavior with respect to the scoring function and a training dataset. In particular, optimality *does not* refer to behavior on unseen testing data.

**Hill climbing with a tabu list and random restarts** (*tabu*). Hill climbing is a widely-used local search technique in discrete optimization (Russell and Norvig, 2003)

that typically finds local optima for an objective function  $f$  by maintaining a *current* solution and applying *search operators*. At each step, all search operators are tentatively applied to the current solution to find its *neighborhood*. The member of the neighborhood which results in the biggest improvement to  $f$  is selected as the new current solution. This process is repeated until a local optimum is found, that is, when the current solution is better than everything in its neighborhood. *Random restarting* is a strategy to escape from a local optimum by randomly changing a locally optimal solution and restarting the search from the new random solution. The *tabu list* strategy (Glover, 1990) augments random restarts by keeping track of recently visited solutions; solutions in the tabu list are ignored when considering new neighborhoods. Even with random restarts and the tabu list, the algorithm is unable to provide guarantees on how close the found local optima are to the globally optimal solutions in terms of their scores.

In the context of Bayesian networks, each solution corresponds to a network; the search operators considered here are edge addition, deletion and reversal (as long as the resulting structure is a DAG). The objective function  $f$  is exactly the scoring function  $s$ .

**Max-min hill climbing (*mmhc*).** Max-min hill climbing (Tsamardinos et al., 2006) is a two-phase hybrid learning algorithm. During the first phase, it uses a set of statistical independence tests to identify arcs that are forbidden from appearing in the learned network. The second phase uses *tabu* to find local optima within this restricted space. Here we use a mutual information statistical test during the first phase. The first phase of *mmhc* is similar to constraint-based methods such as *pc* (Spirtes et al., 2000). Empirically, *mmhc* has been shown to outperform several other state-of-the-art algorithms, including PC, sparse candidate, three phase dependency analysis, optimal reinsertion and greedy equivalence search (Tsamardinos et al., 2006). While *mmhc* does guarantee to recover BN structures when the data are faithful to a DAG in the large sample limit (Tsamardinos et al., 2006), it does not offer any non-trivial guarantees about the generalization quality of the learned network for unfaithful, finite datasets.

**Chow-Liu (*cl*).** The Chow-Liu algorithm (Chow and Liu, 1968) is an exact, polynomial-time algorithm for finding an optimal tree-structured BN. The algorithm calculates the mutual information between all pairs of variables to form a weighted graph. The maximum spanning tree through the graph corresponds to the optimal tree-structured BN.

**Provably optimal (*opt*).** Several algorithms have been developed which are guaranteed to find a network which optimizes  $s$  (Ott and Miyano, 2003; Koivisto and Sood, 2004; Silander and Myllymäki, 2006; de Campos and Ji, 2011; Parviainen and Koivisto, 2009; Cussens, 2011; Yuan and Malone, 2013). In practice, these algorithms take as input

a set of local scores for each variable and find an optimal network with respect to these scores. In this work, we use two of these algorithms which have previously (Malone et al., 2014) been shown to perform well on a variety of datasets. The first (Yuan and Malone, 2013) is based on casting BNSL as a shortest-path finding problem; it then uses  $A^*$  to solve the shortest path problem, which gives the optimal network for the given local scores. The second algorithm (Bartlett and Cussens, 2013) creates an integer linear program (ILP) based on the local scores. The solution to the ILP corresponds to the optimal network for the local scores. Both  $A^*$  and ILP are guaranteed to find (equivalent) optimal network structures. In this work, our goal is to understand the impact of the optimality guarantee on generalization. Since we are not interested in the relative performance in terms of running time or memory consumption among the algorithms, we make no distinction between the different optimal algorithms.

## 4 QUALITY MEASURES

In order to address our research question Q1, we propose a normalized version of the structural Hamming distance to quantify structural similarity. Research questions Q2–Q4 concern generalization; we propose measures based on the predictive likelihood of Equation 1 to evaluate these results.

### 4.1 Structural Similarity

We evaluate the structural similarity of two networks with structural Hamming distance (SHD) (Tsamardinos et al., 2006). The SHD between two networks is calculated by transforming the two networks into the partially directed acyclic graphs (PDAGs) representing their equivalence classes. The number of edge additions, deletions, reversals, and orientation changes (converting an undirected edge into a directed edge and vice versa) to transform one PDAG into the other is the SHD. The motivation behind SHD lies in equivalence classes of BNs; using different conditional probability distributions, different DAG structures can describe exactly the same set of joint probability distributions (Chickering, 1995); these DAGs belong to the same *equivalence class*. The SHD is 0 between DAGs in the same equivalence class. Thus, in a sense, SHD serves as an imperfect proxy for measuring the distance between the distributions represented by two DAGs. In order to facilitate comparison across datasets with differing numbers of variables, we use a normalized form  $\widehat{SHD}$  of SHD:

$$\widehat{SHD}_d = \frac{SHD_d}{\binom{n_d}{2}/2}, \quad (2)$$

where  $SHD_d$  is the structural Hamming distance between two networks for dataset  $d$ ,  $n_d$  is the number of variables in dataset  $d$  and  $\binom{n_d}{2}$  is the binomial coefficient. The normalization constant  $\binom{n_d}{2}/2$  is approximately the maximum

number of edges present in a network structure, and hence also in the order of the SHD between two networks learned for dataset  $d$ .

## 4.2 Predictive Likelihood

We use the predictive likelihood to evaluate the generalization capability of the learned networks. In particular, for a dataset  $d$  and learning strategy  $l$  we calculate the per-prediction-likelihood,  $\ell_{pp}^{d,l}$ , which is the likelihood of each prediction on the test set:

$$\ell_{pp}^{d,l} = -\frac{\sum_i^{10} \ell_i^{d,l}}{N_d \cdot n_d}, \quad (3)$$

summing over the folds  $i = 1..10$ , where  $\ell_i^{d,l}$  is the predictive likelihood according to Equation 1 on the test set for fold  $i$  using learning strategy  $l$  (see Section 5.1 for cross-validation discussion),  $N_d$  is the number of records in the test set, and  $n_d$  is the number of variables in the dataset.

The numerator of Equation 3 is the sum over all of the test set predictive likelihoods for learning strategy  $l$  and dataset  $d$ . As discussed in Section 2, each  $\ell_i^{d,l}$  term comprises  $\frac{N_d}{10} \cdot n_d$  terms. In total, the sum in the numerator includes  $N_d \cdot n_d$  terms, each of which corresponds to the log probability of one variable of one record from the test set. Consequently, the denominator serves as a normalizing constant, and  $\ell_{pp}^{d,l}$  is the average log probability of each prediction.

In order to compare learning strategies, we normalize the  $\ell_{pp}^{d,l}$  values for each dataset between 0 and 1 to obtain

$$\hat{\ell}_{pp}^{d,l} = 1 - \frac{\ell_{pp}^{d,l} - \min_{l'} \{\ell_{pp}^{d,l'}\}}{\max_{l'} \{\ell_{pp}^{d,l'}\} - \min_{l'} \{\ell_{pp}^{d,l'}\}} \quad (4)$$

where  $l'$  ranges over all learning strategies. Note that, after normalization, the learning strategy with the best  $\ell_{pp}^{d,l}$  has  $\hat{\ell}_{pp}^{d,l} = 0$  while the worst learning strategy has  $\hat{\ell}_{pp}^{d,l} = 1$ .

It is important to note that  $\ell_{pp}^{d,l}$  and  $\hat{\ell}_{pp}^{d,l}$  consider all variables equally. In particular, they do not consider a special “class” variable.

## 5 EXPERIMENTAL SETUP

We continue by describing the experiment setup.

### 5.1 Datasets

We used datasets from a previous wide-scale empirical study that focused on predicting the efficiency of BNSL algorithms (Malone et al., 2014)<sup>1</sup>. In total, we obtained

<sup>1</sup>The datasets are available at <http://bnportfolio.cs.helsinki.fi/>. Please see the original study for data pre-processing.

Table 1: Basic dataset characteristics

| Dataset                     | Type       | $n$ | $N$    |
|-----------------------------|------------|-----|--------|
| agaricus                    | <i>uci</i> | 22  | 8 123  |
| alarm <sub>10 000</sub>     | <i>sam</i> | 37  | 10 000 |
| alarm <sub>1 000</sub>      | <i>sam</i> | 37  | 1 000  |
| alarm <sub>100</sub>        | <i>sam</i> | 37  | 100    |
| carpo <sub>10 000</sub>     | <i>sam</i> | 60  | 10 000 |
| carpo <sub>1 000</sub>      | <i>sam</i> | 58  | 1 000  |
| carpo <sub>100</sub>        | <i>sam</i> | 56  | 100    |
| connect <sub>6 000</sub>    | <i>sam</i> | 39  | 6 000  |
| anneal                      | <i>uci</i> | 32  | 897    |
| credit                      | <i>uci</i> | 18  | 1 000  |
| lymph                       | <i>uci</i> | 19  | 147    |
| tumor                       | <i>uci</i> | 18  | 338    |
| dermatology                 | <i>uci</i> | 34  | 365    |
| flag                        | <i>uci</i> | 27  | 193    |
| hailfinder <sub>1 000</sub> | <i>sam</i> | 56  | 1 000  |
| hailfinder <sub>100</sub>   | <i>sam</i> | 56  | 100    |
| votes                       | <i>uci</i> | 17  | 434    |
| hypothyroid                 | <i>uci</i> | 22  | 3 771  |
| insurance <sub>10 000</sub> | <i>sam</i> | 27  | 10 000 |
| kredit                      | <i>uci</i> | 18  | 1 000  |
| kr-vs-kp                    | <i>uci</i> | 37  | 3 195  |
| letter                      | <i>uci</i> | 17  | 20 000 |
| lung                        | <i>uci</i> | 57  | 31     |
| mildew <sub>1 000</sub>     | <i>sam</i> | 35  | 1 000  |
| mildew <sub>100</sub>       | <i>sam</i> | 35  | 100    |
| soybean                     | <i>uci</i> | 36  | 306    |
| spect                       | <i>uci</i> | 23  | 186    |
| water <sub>100</sub>        | <i>sam</i> | 26  | 100    |
| zoo                         | <i>uci</i> | 17  | 100    |

29 datasets from two categories used in that study: 16 are “real” datasets from the UCI Machine Learning Repository (*uci*), and further 13 are generated using logic sampling from well-known benchmark networks (*sam*).

As Table 1 shows, the number of variables in the datasets ranges from 17 to 60, and the number of records ranges from about 30 to 20 000. We used standard 10-fold cross-validation in order to evaluate the learning strategies. Datasets were randomly split into 10 folds. Unless otherwise noted, all results are averages over all 10 folds. For a few datasets, not all learning strategies completed on all folds. In these cases, the averages were adjusted to properly account for the number of completed folds.

### 5.2 Learning and Inference

Most of the learning algorithm implementations are publicly available. In all cases, default parameter values were used (excluding number of parents).

**Exact algorithms.** The previous study (Malone et al., 2014) found that the ILP algorithm usually ran faster than A\* for datasets with up to 10 000 local scores. In order to limit the computational requirements of this study, we used this simple decision rule to select either ILP or A\* for each dataset. We used a time limit of 2 hours. If the selected algorithm failed, the other one was used.

**Parent limit.** For all algorithms except *cl*, we used hard limits of 2 and 8 on the number of parents. This constraint serves two purposes. First, a hard limit on the complexity of the learned network allows evaluation of the learning algorithms in the different search spaces. In particular, the space defined by a parent limit of 2 is a subset of the space defined by a parent limit of 8. Thus, the optimal solution for the at-most-8-parents space is always at least as good as that of the at-most-2-parents space. Second, calculating all local scores for large parent limits is impractical.

Table 2 outlines the seven combinations of learning algorithms and parent limits used in the evaluation.

**Scoring function.** We selected the commonly-used Bayesian Dirichlet with score equivalence and uniform structure prior (BDeu) scoring function (Heckerman et al., 1995) with an equivalent sample size (ESS) of 1 as the scoring function. Note that, while several previous studies (Sillander et al., 2008; Liu et al., 2012; Korucuoglu et al., 2014) have demonstrated that BDeu is sensitive to the ESS, BDeu with an ESS of 1 is a commonly-used score (Acid et al., 2004), which motivates this choice.

**Inference.** For all learned structures, parameter values were set using a symmetric Dirichlet prior with a concentration parameter of 1 (which is equivalent to Laplacian smoothing). All testing likelihood calculations were performed by multiplying relevant family factors.

## 6 RESULTS

In this section, we address each of the research questions Q1–Q4. In Section 6.1, we answer Q1 by showing that  $\widehat{\text{SHD}}$  is typically high among networks learned using different algorithms, particularly for *uci* datasets. In Section 6.2, we show that, surprisingly, the answer to Q2 depends upon learning algorithm and dataset. We address Q3 in Section 6.3, where we find that increasing training data both improves prediction accuracy and reduces variance among cross-validation folds. In Section 6.4 we demonstrate that somewhat unexpectedly, for *sam* datasets simple learning strategies like *cl* perform well. The *opt* strategy consistently generalizes better for *uci* datasets. These observations suggest fundamental differences in the *uci* and *sam* datasets.

### 6.1 Structural Similarity

We address Q1 by evaluating the similarity of learned structures using  $\widehat{\text{SHD}}$  in two different settings. We first consider the similarity of structures learned using the same algorithm on different cross-validation folds. We then investigate the similarity of structures learned using different algorithms on the same cross-validation fold. As a trivial baseline learning “result,” the empty network with no edges, *empty*, is also included. We stress that the goal of Q1

is to evaluate the structural similarity of learned networks to each other; we do not consider e.g. similarity to “gold standard” networks which do not exist for *uci* datasets.

**Variation within a learning algorithm.** Due to the cross-fold validation strategy, each algorithm results in multiple networks on each dataset. We compared the  $\widehat{\text{SHD}}$  among all pairs of networks learned using a single learning strategy. Figure 1(top) shows that, for *sam* datasets, the difference among learned networks was usually small. In contrast, Figure 1(bottom) shows that *uci* datasets result in more varied networks. The variation for *cl* in Figure 1(bottom, right) is much lower than those of either *opt*<sub>8</sub> (left) or *tabu*<sub>2</sub> (center). This highlights how the reduced search space decreases variability among optimal structures.

Interestingly, for *uci* datasets, *opt*<sub>8</sub> networks tend to have a slightly higher  $\widehat{\text{SHD}}$  than those of *tabu*<sub>2</sub>. However, the variance for *opt*<sub>8</sub> is smaller than that of *tabu*<sub>2</sub>. One interpretation is that *opt*<sub>8</sub> networks are, in terms of  $\widehat{\text{SHD}}$ , “equally spaced.” On the other hand, some pairs of the *tabu*<sub>2</sub> networks are quite similar, while others have more pronounced differences. An explanation for this phenomenon is the greedy search strategy of *tabu*<sub>2</sub>. In the beginning stages of the search, *tabu*<sub>2</sub> is likely to select the same dominant edges, regardless of the nuances of the dataset at hand. In contrast, *opt*<sub>8</sub> selects accurate substructures and optimally combines them, so it may disregard single strong edges in favor of more informative structures. Another explanation is that the search space for *tabu*<sub>2</sub> is more constrained than that of *opt*<sub>8</sub>. As further evidence for the more constrained space leading to more similar networks, both *cl* (Figure 1(bottom, right)) and *opt*<sub>2</sub> on *uci* datasets (not shown) typically result in smaller  $\widehat{\text{SHD}}$  than *tabu*<sub>2</sub>.

**Variation between learning algorithm.** We additionally compared the average  $\widehat{\text{SHD}}$  among networks learned using different strategies on the same training set. Figure 2 shows that some of the strategies learn quite similar networks, while for other strategies the networks differ quite a bit. Perhaps unsurprisingly, the same learning algorithm with different parent limits often result in similar networks. The *mmhc* networks are quite similar to *empty*, which suggests that they are very sparse. Many of the datasets have fewer than 1 000 records. So the statistical tests employed at the beginning of *mmhc* are often unable to detect dependencies, and thus many possible edges are discarded before beginning the score-based search.

In general, all of the learning strategies tend to learn similar networks for the *sam* datasets, while the *uci* datasets result in more diverse structures. Despite this, some patterns among the pairs of learning strategies are consistent among both types of datasets. For example, *opt*<sub>8</sub> and *cl* exhibit the highest average  $\widehat{\text{SHD}}$ . On the other hand, *tabu*<sub>2</sub> and *opt*<sub>2</sub> are more similar than most other pairs for the *uci* datasets, but not for *sam* datasets. This again suggests that the more



Table 2: Learning algorithms used in the study

| Algorithm   | Parent limits | Abbreviation                                      | Availability   |
|-------------|---------------|---|--|
| <i>tabu</i> | 2, 8          | <i>tabu<sub>2</sub></i> , <i>tabu<sub>8</sub></i> | <a href="http://www.bnlearn.com/">http://www.bnlearn.com/</a>  |
| <i>mmhc</i> | 2, 8          | <i>mmhc<sub>2</sub></i> , <i>mmhc<sub>8</sub></i> | <a href="http://www.bnlearn.com/">http://www.bnlearn.com/</a>  |
| <i>cl</i>   | -             | <i>cl</i>   | custom   |
| <i>opt</i>  | 2, 8          | <i>opt<sub>2</sub></i> , <i>opt<sub>8</sub></i>   | <a href="http://urlearning.org">http://urlearning.org</a><br><a href="http://www.cs.york.ac.uk/aig/sw/gobnilp/">http://www.cs.york.ac.uk/aig/sw/gobnilp/</a> |

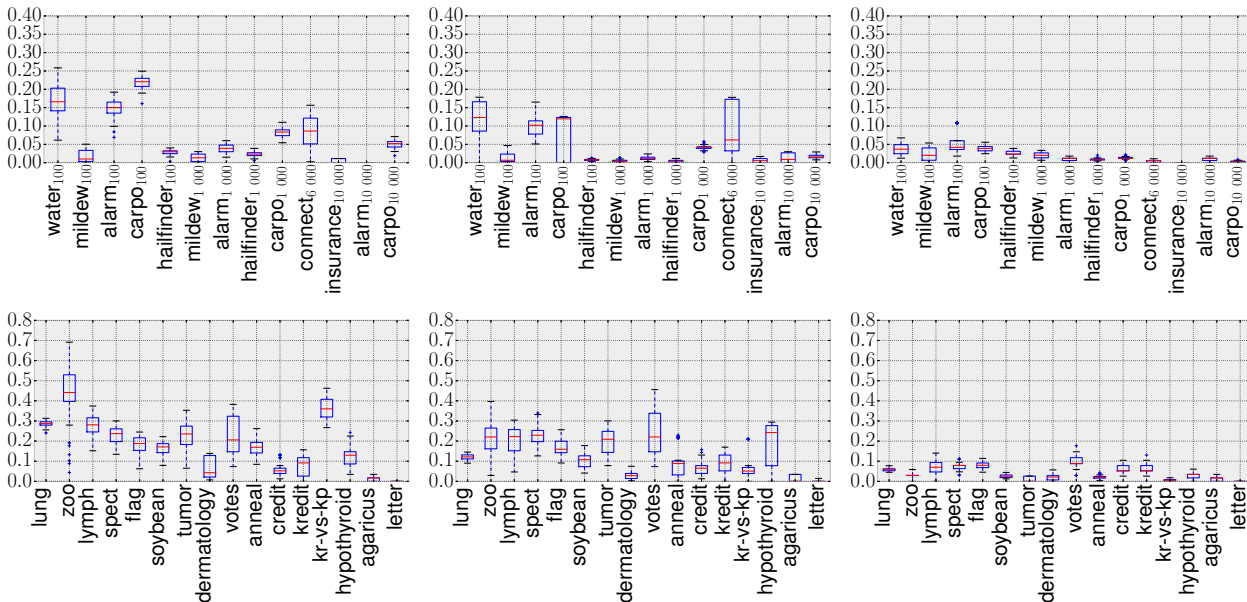


Figure 1: The  $\widehat{SHD}$  values for *opt<sub>8</sub>* (left), *tabu<sub>2</sub>* (center) and *cl* (right). The top row contains *sam* datasets, and the bottom row contains *uci* datasets. The datasets are sorted in ascending number of records. Note the different scale for *sam* and *uci*.

constrained search space leads to more similar structures.

In summary for Q1, the similarity of learned network structures depends upon the nature of the training dataset. Typically, though, both within and between learning algorithms, *sam* datasets result in similar learned structures, while structures learned from *uci* datasets are more diverse.

## 6.2 Impact of Restricting Parent Set Size

We study question Q2 by comparing the  $\hat{\ell}_{pp}^{d,l}$  among datasets when using  $k = 2$  and  $k = 8$  as the maximum number of parents for each learning algorithm. The BDeu score implicitly restricts the maximum number of selected parents as a soft constraint by integrating over all parameterizations of parent instantiations. Other scores, such as MDL, explicitly incorporate a complexity penalty to discourage large parent sets. In both cases, though, this restriction is a soft constraint. Here consider the maximum number of parents as a *hard constraint*.

**Optimal.** Figure 3 (left) shows the performance (in terms of  $\hat{\ell}_{pp}^{d,l}$ ) of generalization using *opt<sub>k</sub>* for parent limits  $k = 2, 8$ . The (left, top) and (left, bottom) plots show distinctly

different patterns. Figure 3 (left, top) clearly shows that *opt<sub>2</sub>* results in better generalization for *sam* datasets with 100 records. However, as the number of records increases, *opt<sub>8</sub>* yields better performance. In contrast, for *uci* datasets, *opt<sub>8</sub>* is almost always better.

**Tabu.** Contrasting the results for *opt*, Figure 3 (center, bottom) shows that *tabu<sub>2</sub>* generalizes better than *tabu<sub>8</sub>* for *uci* datasets. One possible explanation for this difference is that the greedy strategy of *tabu<sub>8</sub>* favors structures which improve the likelihood while increasing the complexity of the learned structures. Thus, the learned structure overfits the training data and does not generalize well to testing data. In contrast, as *opt* is guaranteed to find the best-scoring structure, it finds structures which better balance training set likelihood and complexity. The hard constraints on the number of parents for *tabu<sub>2</sub>* forbid it from selecting the complex structures. Both *tabu<sub>2</sub>* and *tabu<sub>8</sub>* typically generalize well on *sam* datasets.

**MMHC.** Figure 3 (right) shows that the hard parent limit has little effect on  $\hat{\ell}_{pp}^{d,l}$  for *mmhc*. The first phase of *mmhc* uses a set of statistical independence tests to restrict the learned network structures. For many of the datasets, the

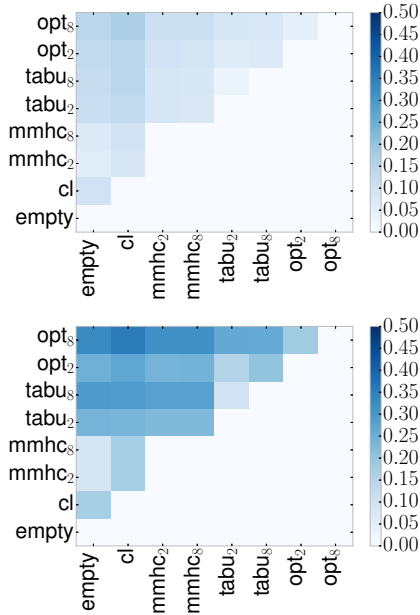


Figure 2: The average  $\widehat{\text{SHD}}$  values between networks learned using the same training set among all algorithms for *sam* (top) and *uci* (bottom) datasets. Lighter colors indicate more similar structures. The lower triangle is left empty due to  $\widehat{\text{SHD}}$  being symmetric.

relatively small number of records restricts the power of these tests and leads to a very small search space in the second phase, despite initially allowing many more structures for the 8-parent space.

Based on these observations, in the rest of this study we will focus on  $opt_8$ ,  $tabu_2$  and  $mmhc_8$ .

In summary, the answer to Q2 clearly depends both on the training datasets and learning algorithm; the global guarantees of  $opt$  allow it to fully take advantage of the larger  $k = 8$  search space, but the local search strategy of  $tabu$  performs better in the more restricted  $k = 2$  space.

More data is required to accurately estimate the conditional probability distributions for complex structures (with more parameters). This may explain why  $opt_2$  generalizes better than  $opt_8$  for datasets with a small number of records.

### 6.3 Impact of Amount of Training Data

To investigate the impact of the amount of available training data, to answer Q3 we compared how  $\ell_{pp}^{d,l}$  of  $opt_8$ ,  $tabu_2$  and  $cl$  behave as the number of records available for training increases. Figure 4 shows that for all algorithms on both *sam* and *uci* datasets, more records lead to better  $\ell_{pp}^{d,l}$ . Furthermore, the plots also show that with more records, the variance of  $\ell_{pp}^{d,l}$  decreases. Interestingly, the plot also shows that  $cl$  performs better than  $opt_8$  and  $tabu_2$  on *carpo*, a *sam* dataset, when only 100 records are available. This again

highlights that restricted model classes can generalize better than those with more parameters, especially when little data is available to estimate the parameter values. Despite the differences in guarantees,  $opt_8$ ,  $tabu_2$  and  $cl$  perform similarly for *carpo*<sub>1 000</sub> and *carpo*<sub>10 000</sub>.

As with *carpo*, for the *uci agaricus* dataset, the likelihood improves and variance decreases as the number of records increases. However,  $opt_8$  improves from  $\ell_{pp}^{d,l} \approx 0.7$  for 81 records to  $\ell_{pp}^{d,l} \approx 0.48$  with 812 records. In contrast,  $tabu_2$  only improves from  $\ell_{pp}^{d,l} \approx 0.7$  to about  $\ell_{pp}^{d,l} \approx 0.55$ , and  $cl$  exhibits even less improvement. For *agaricus*,  $opt_8$  using only 812 records results in better generalization than  $tabu_2$  or  $cl$  with all 8 123 records.

We observed similar behavior on other *sam* and *uci* datasets as the amount of training data was varied. Unlike in the case of Q1 and Q2, the same general trends hold for all algorithms and datasets with respect to Q3. Namely, the predictive likelihood improves and variance decreases as the size of the training set increases.

### 6.4 Comparison Across Learning Strategies

Finally, based on the previous results, we studied Q4 by choosing the best learning strategies and comparing their  $\hat{\ell}_{pp}^{d,l}$  across all of the datasets. In essence, we fix the training set while varying the learning strategy. Additionally, *empty* (with no edges) was included as a baseline. The results in Figure 5 show several expected trends and a few surprises. As expected, *empty* is the worst on almost all of the datasets. Due to its structural similarity to *empty*,  $mmhc_8$  was typically worse than the other strategies. These trends are consistent for both *sam* and *uci* datasets. For *sam* datasets,  $tabu_2$  and  $opt_8$  have very similar  $\hat{\ell}_{pp}^{d,l}$  for most datasets; the  $\hat{\ell}_{pp}^{d,l}$  of  $cl$  is also surprisingly similar to that of the two more “sophisticated” strategies.

For *uci* datasets,  $opt_8$  continues to consistently have good  $\hat{\ell}_{pp}^{d,l}$ . On the other hand,  $cl$  and  $tabu_2$  exhibit much more inconsistency in their generalization relative to  $opt_8$ . For some datasets, such as *dermatology* and *kredit*, they match  $opt_8$ ; on others, such as *credit* and *tumor*,  $cl$  and  $tabu_2$  do not generalize well. Surprisingly,  $cl$  exhibits the best  $\hat{\ell}_{pp}^{d,l}$  for *letter*, the *uci* dataset with the most records.

For Q4,  $opt$  guarantees consistently translate into networks with good generalization. Algorithms with weaker guarantees produce networks with inconsistent generalization.

**Comments on Datasets.** Besides the behavior of the learning algorithms, these results also suggest differences in the datasets themselves. In particular, it seems that *sam* datasets are “easier,” in the sense that many learning strategies find networks which generalize well. On the other hand, only the strategy with strong guarantees consistently generalizes well on *uci* datasets. In some sense, this result is not surprising. The *sam* data is by construction accurately

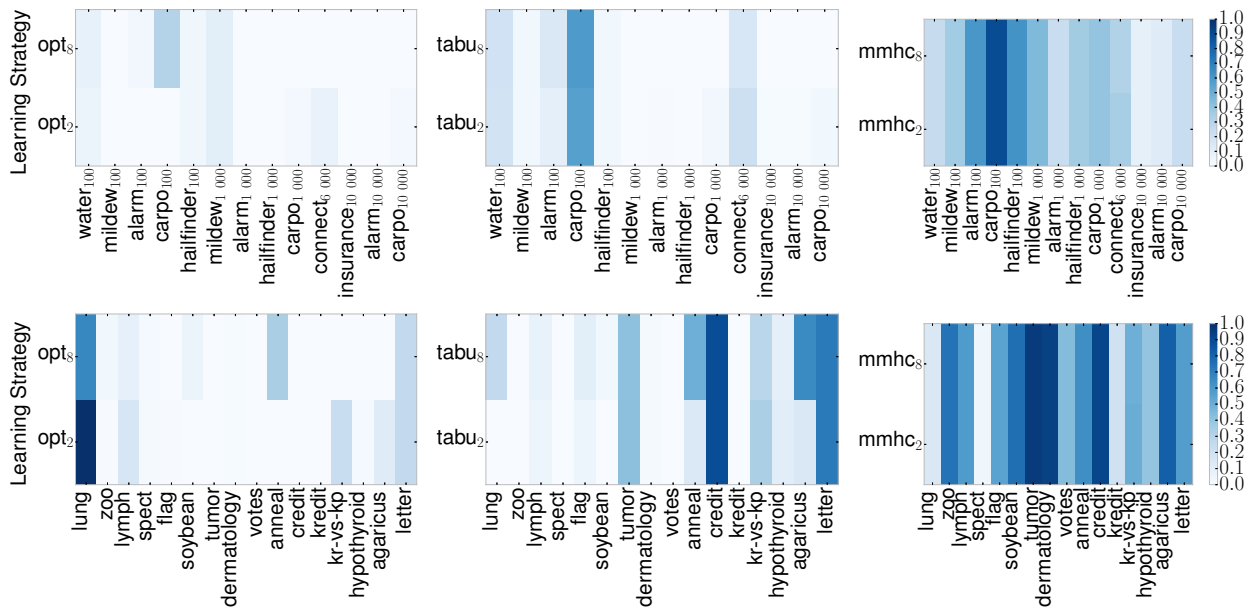


Figure 3: The  $\hat{\ell}_{pp}^{d,l}$  values for  $opt$  (left),  $tabu$  (center) and  $mmhc$  (right) with a hard limit of  $k = 2$  and  $k = 8$  for  $sam$  (top) and  $uci$  (bottom) datasets. The datasets are sorted in ascending number of records. Lighter colors indicate better performance. Close inspection of the  $mmhc$  strategies show some slight difference; however, they are difficult to discern in the scaled image.

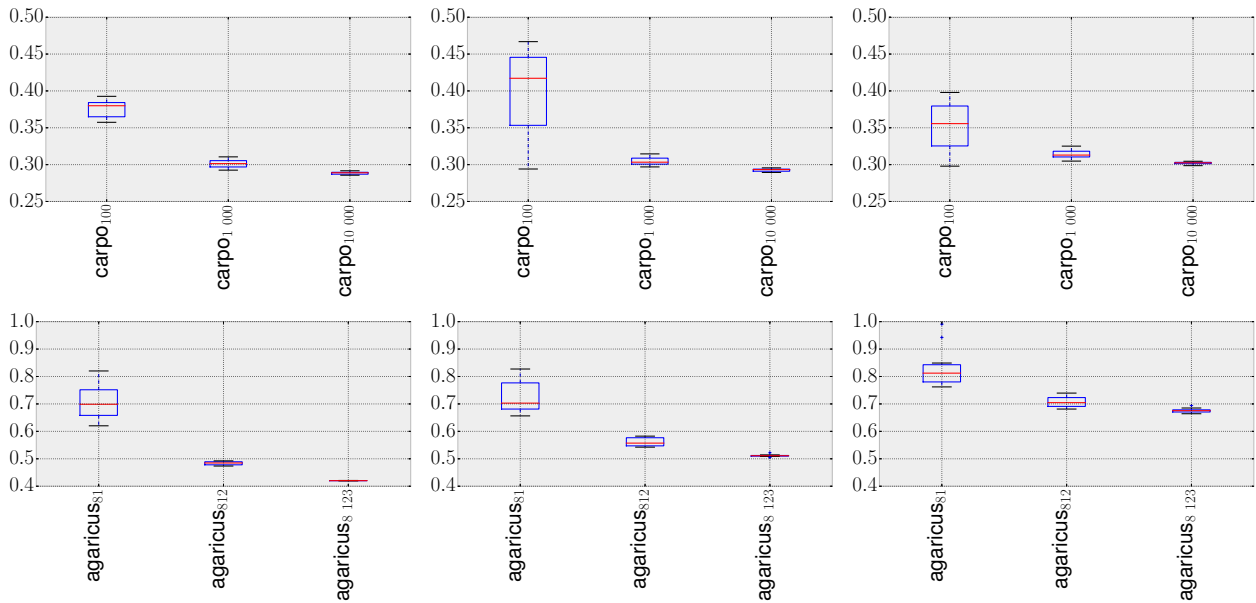


Figure 4: The  $\hat{\ell}_{pp}^{d,l}$  values for using the  $opt_8$  (left),  $tabu_2$  (center), and  $cl$  (right) learning strategies as the number of records increases. The top row is for the  $carpo$  dataset ( $sam$ ); the bottom row is for the  $agaricus$  dataset ( $uci$ ). Note the different y-axes for the plots. Lower values and smaller boxes are better.

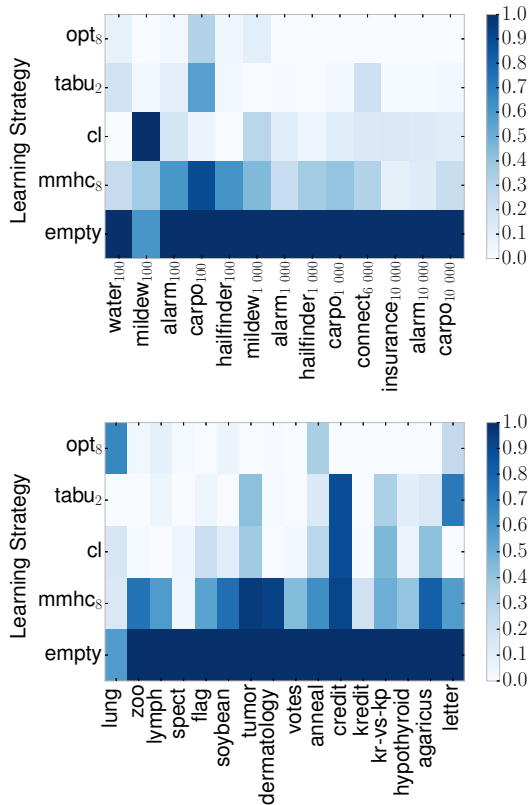


Figure 5: The  $\hat{\rho}_{pp}^{d,l}$  values for the best learning strategies. The empty network is included as a baseline. The *sam* datasets are shown in the top heatmap, and *uci* datasets are in the bottom. The datasets are sorted in ascending number of records. Lighter colors indicate better performance.

modeled by a BN, while it is very unlikely that *uci* datasets are faithful to any BN. These caveats are also important for future evaluations.

## 7 CONCLUSIONS

In this work, we systematically evaluated the impact of learning strategy, including choice of learning algorithm and hard constraints on the number of parents for variables in the network, on the properties of the learned network. Our experiments address the four research questions introduced in Section 1. In particular, we answered Q1 by showing that different learning strategies result in dissimilar structures, particularly for *uci* datasets. For Q2, we showed that for small *sam* datasets, *opt* generalizes better when constrained to 2 parents; however, for all other datasets considered in this study, *opt*<sub>8</sub> performed better. In contrast, *tabu*<sub>2</sub> almost always outperformed *tabu*<sub>8</sub>. Q3 had the clearest answer of the four questions; increasing the size of the training set consistently both improved the predictive likelihood and decreased its variance across cross-validation

folds. Finally, with respect to Q4, for the datasets in this evaluation, *opt* consistently results in networks with good generalization. Nevertheless, for some of the *sam* datasets, simpler strategies such as the polynomial-time Chow-Liu algorithm yielded nearly as good generalization. In our view, these results justify the research into learning optimal BN structures in large, complex spaces.

The aim of the study was to better understand how the combination of learning strategy and dataset affect generalization. Consequently, we deliberately disregarded the runtime and memory usage of the studied learning strategies. These are important constraints, especially for *opt*, i.e., exact algorithms which provide provably optimal network structures with respect to the score-based objective function. However, the results clearly show that, whenever possible (as long as the computational resources allow it), it is worthwhile to use *opt*. Similarly, we did not evaluate the SHD between learned network structures and a “gold standard” network because this does not directly reflect generalization. Also, trustworthy “gold standard” networks do not generally exist for real-world datasets.

This empirical study clarifies some common assumptions about relationship between structure learning algorithms and learned Bayesian network structures, including empirically observed conditions for strong theoretical guarantees translating into improved empirical results. The results suggest many interesting questions for further study. For example, the finding that quite simple networks can generalize well suggests that a scoring function with a high complexity penalty, such as the Bayesian Information Criterion (BIC), might yield networks with good predictive capabilities. Another interesting question concerns the impact of more sophisticated restrictions on learned networks structures. For example, recently several algorithms (Korhonen and Parviainen, 2013; Berg et al., 2014; Parviainen et al., 2014) have been proposed which find provably optimal BNs with bounded treewidth, resulting in networks for which exact inference is provably tractable. As bounded treewidth represents another well-principled approach to constraining complexity of the learned networks, an interesting question is whether the quality of the learned (optimal) networks is affected by such a stringent constraint. Other extensions of this work would be to involve yet more structure learning algorithms (such as the Greedy Equivalence Search (Chickering, 2002)), and extending from using single structure for predictive inference to a more Bayesian approach by collecting several high-scoring networks and averaging their predictions. Additionally, the predictive likelihood analysis is not restricted to Bayesian networks; extensions to other generative models, like Markov random fields, would also be of interest.

**Acknowledgements** This research was supported in part by the Academy of Finland (grants 251170/COIN, 276412, and 284591).

## References

- Acid, S., de Campos, L. M., Fernandez-Luna, J. M., Rodriguez, S., Rodriguez, J. M., and Salcedo, J. L. (2004). A comparison of learning algorithms for Bayesian networks: a case study based on data from an emergency medical service. *Artificial Intelligence in Medicine*, 30(3):215–232.
- Bartlett, M. and Cussens, J. (2013). Advances in Bayesian network learning using integer programming. In *Proc. UAI*, pages 182–191. AUAI Press.
- Berg, J., Järvisalo, M., and Malone, B. (2014). Learning optimal bounded treewidth Bayesian networks via maximum satisfiability. In *Proc. AISTATS*, pages 86–95. JMLR.org.
- Chickering, D. M. (1995). A transformational characterization of equivalent Bayesian network structures. In *Proc. UAI*, pages 87–98. Morgan Kaufmann.
- Chickering, D. M. (1996). Learning Bayesian networks is NP-complete. In *Learning from Data: Artificial Intelligence and Statistics V*, pages 121–130. Springer-Verlag.
- Chickering, D. M. (2002). Learning equivalence classes of Bayesian-network structures. *Journal of Machine Learning Research*, 2:445–498.
- Chow, C. and Liu, C. (1968). Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467.
- Cussens, J. (2011). Bayesian network learning with cutting planes. In *Proc. UAI*, pages 153–160. AUAI Press.
- de Campos, C. P. and Ji, Q. (2011). Efficient learning of Bayesian networks using constraints. *Journal of Machine Learning Research*, 12:663–689.
- de Jongh, M. and Druzdzel, M. J. (2009). A comparison of structural distance measures for causal Bayesian network models. In *Recent Advances in Intelligent Information Systems, Challenging Problems of Science, Computer Science series*, pages 443–456. Academic Publishing House EXIT.
- Friedman, N., Nachman, I., and Peer, D. (1999). Learning Bayesian network structure from massive datasets: The “sparse candidate” algorithm. In *Proc. UAI*, pages 206–215. Morgan Kaufmann.
- Glover, F. (1990). Tabu search: A tutorial. *Interfaces*, 20(4):74–94.
- Heckerman, D., Geiger, D., and Chickering, D. M. (1995). Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20:197–243.
- Koivisto, M. and Sood, K. (2004). Exact Bayesian structure discovery in Bayesian networks. *Journal of Machine Learning Research*, 5:549–573.
- Korhonen, J. H. and Parviainen, P. (2013). Exact learning of bounded tree-width Bayesian networks. In *Proc. AISTATS*, pages 370–378. JMLR.org.
- Korucuoglu, M., Isci, S., Ozgur, A., and Otu, H. H. (2014). Bayesian pathway analysis of cancer microarray data. *PLoS ONE*, 9(7):e102803.
- Liu, Z., Malone, B., and Yuan, C. (2012). Empirical evaluation of scoring functions for Bayesian network model selection. *BMC Bioinformatics*, 13(Suppl 15):S14.
- Malone, B., Kangas, K., Järvisalo, M., Koivisto, M., and Myllymäki, P. (2014). Predicting the hardness of learning Bayesian networks. In *Proc. AAAI*, pages 2460–2466. AAAI Press.
- Mitchell, T. (1997). *Machine Learning*. McGraw-Hill.
- Ott, S. and Miyano, S. (2003). Finding optimal gene networks using biological constraints. *Genome Informatics*, 14:124 – 133.
- Parviainen, P., Farahani, H. S., and Lagergren, J. (2014). Learning bounded tree-width Bayesian networks using integer linear programming. In *Proc. AISTATS*, pages 751–759. JMLR.org.
- Parviainen, P. and Koivisto, M. (2009). Exact structure discovery in Bayesian networks with less space. In *Proc. UAI*, pages 436–443. AUAI Press.
- Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers Inc.
- Russell, S. J. and Norvig, P. (2003). *Artificial Intelligence: A Modern Approach*. Pearson Education.
- Silander, T. and Myllymäki, P. (2006). A simple approach for finding the globally optimal Bayesian network structure. In *Proc. UAI*, pages 445–452. AUAI Press.
- Silander, T., Roos, T., Kontkanen, P., and Myllymäki, P. (2008). Factorized normalized maximum likelihood criterion for learning Bayesian network structures. In *Proc. PGM*, pages 257–272.
- Spirites, P., Glymour, C., and Schemes, R. (2000). *Causation, Prediction, and Search*. The MIT Press, 2 edition.
- Teyssier, M. and Koller, D. (2005). Ordering-based search: A simple and effective algorithm for learning Bayesian networks. In *Proc. UAI*, pages 548–549. AUAI Press.
- Tsamardinos, I., Brown, L., and Aliferis, C. (2006). The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65:31–78.
- Ueno, M. (2010). Learning networks determined by the ratio of prior and data. In *Proc. UAI*, pages 598–605. AUAI Press.
- Ueno, M. (2011). Robust learning Bayesian networks for prior belief. In *Proc. UAI*, pages 698–707. AUAI Press.
- Yuan, C. and Malone, B. (2013). Learning optimal Bayesian networks: A shortest path perspective. *Journal of Artificial Intelligence Research*, 48:23–65.

---

# Learning the Structure of Causal Models with Relational and Temporal Dependence

---

**Katerina Marazopoulou**  
kmarazo@cs.umass.edu

**Marc Maier**  
maier@cs.umass.edu  
College of Information and Computer Sciences  
University of Massachusetts Amherst  
Amherst, MA 01003

**David Jensen**  
jensen@cs.umass.edu

## Abstract

Many real-world domains are inherently *relational* and *temporal*—they consist of heterogeneous entities that interact with each other over time. Effective reasoning about causality in such domains requires representations that explicitly model relational and temporal dependence. In this work, we provide a formalization of temporal relational models. We define temporal extensions to abstract ground graphs—a lifted representation that abstracts paths of dependence over all possible ground graphs. Temporal abstract ground graphs enable a sound and complete method for answering *d*-separation queries on temporal relational models. These methods provide the foundation for a constraint-based algorithm, TRCD, that learns causal models from temporal relational data. We provide experimental evidence that demonstrates the need to explicitly represent time when inferring causal dependence. We also demonstrate the expressive gain of TRCD compared to earlier algorithms that do not explicitly represent time.

## 1 INTRODUCTION

Recent work in artificial intelligence has devoted increasing attention to learning and reasoning with causal knowledge. Causality is central to understanding the behavior of complex systems and to selecting actions that will achieve particular outcomes. Thus, causality is implicitly or explicitly central to mainstream AI areas such as planning, cognitive modeling, computational sustainability, game playing, multiagent systems, and robotics. Causal inference is also central to many areas beyond AI, including medicine, public policy, and nearly all areas of science.

Substantial research advances have been made over the past several decades that allow the structure and param-

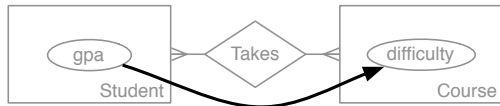
eters of causal graphical models to be learned from observational data. As early as the 1990s, researchers developed constraint-based algorithms, such as the IC [Pearl and Verma, 1991] and PC [Spirtes *et al.*, 2000] algorithms, that leverage the connection between the graphical criterion of *d*-separation and the conditional independencies that are inferred to hold in the underlying distribution, in order to learn the structure of causal graphical models from data.

In this work, we significantly extend the expressiveness of the models learnable with constraint-based algorithms. Specifically, we provide a formalization of temporal relational models, an expressive class of models that can capture probabilistic dependencies between variables on different types of entities within and across time points. We extend the notion of abstract ground graphs [Maier *et al.*, 2013b]—a lifted representation that allows reasoning about the conditional independencies implied by a relational model—for temporal relational models, and we show that temporal abstract ground graphs are a sound and complete abstraction for ground graphs of temporal relational models. Temporal abstract ground graphs can be used to answer *d*-separation queries for temporal relational models. We also extend an existing constraint-based algorithm for inferring causal dependence in relational data—the relational causal discovery (RCD) algorithm—to incorporate time, thus providing a constraint-based method that learns causal models from temporal relational data.

## 2 RELATIONAL MODELS

Propositional representations, such as Bayesian networks, describe domains containing a single entity class. Many real world systems comprise instances from multiple entity classes whose variables are interdependent. Such domains are often referred to as *relational*. In this section, we introduce basic concepts of relational representations that exclude time.

A *relational schema*  $\mathcal{S} = (\mathcal{E}, \mathcal{R}, \mathcal{A}, \text{card})$  specifies the types of entities, relationships, and attributes that exist in a domain. It includes a cardinality function that constrains



$[Course, Takes, Student].gpa \rightarrow [Course].difficulty$

Figure 1: Relational model for the example domain. The underlying relational schema (ER diagram) is shown in gray.

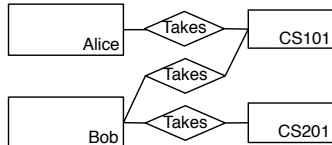


Figure 2: Example relational skeleton for the schema shown in Figure 1.

the number of times an entity instance can participate in a relationship. A relational schema can be depicted with an Entity-Relationship (ER) diagram. Figure 1 shows an example ER diagram (in gray) that describes a simplified university domain. The domain consists of two entity classes (*Student* and *Course*), and one relationship class (*Takes*). The entity class *Student* has one attribute, *gpa*, and the entity class *Course* has one attribute, *difficulty*. The cardinality constraints are shown with crow’s feet notation—students can enroll in multiple courses and a course can be taken by multiple students. A *relational skeleton* is a partial instantiation of a relational schema. It specifies the entity and relationship instances that exist in the domain. Figure 2 shows an example relational skeleton for the relational schema of Figure 1. The skeleton consists of two *Student* instances, Alice and Bob, and two *Course* instances, CS101 and CS201. Alice is taking CS101 and Bob is taking both courses.

Given a relational schema, we can specify *relational paths*, which intuitively correspond to ways of traversing the schema. For the schema shown in Figure 1, possible paths include  $[Student, Takes, Course]$  (the courses a student takes), as well as  $[Student, Takes, Course, Takes, Student]$  (other students that take the same courses). *Relational variables* consist of a relational path and an attribute that can be reached through that path. For example, the relational variable  $[Student, Takes, Course].difficulty$  corresponds to the difficulty of the courses that a student takes. Probabilistic dependencies can be defined between relational variables. Dependencies are said to be in canonical form when the path of the effect (or *outcome*) relational variable is a single item. For canonical dependencies, the path of the cause (or *treatment*) relational variable describes how dependence is induced. As an example, consider the

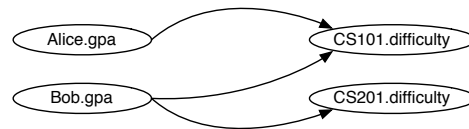


Figure 3: Ground graph for the model of Figure 1 applied on the relational skeleton of Figure 2.

following *relational dependency*

$$[Course, Takes, Student].gpa \rightarrow [Course].difficulty$$

which states that the *difficulty* of a course is affected by the *gpa* of students taking that course. Presumably, instructors adjust the difficulty of the course based on the grade-point average of enrolled students.

A *relational model*  $\mathcal{M} = (\mathcal{S}, \mathcal{D}, \Theta)$  is a collection of relational dependencies defined over a single relational schema along with their parameterizations (a conditional probability distribution for each attribute given its parents). The structure of a relational model can be depicted by superimposing the dependencies on the ER diagram of the relational schema, as shown in Figure 1, and labeling each arrow with the corresponding relational dependency.

Given a relational model  $\mathcal{M}$  and a relational skeleton  $\sigma$ , we can construct a *ground graph*  $GG_{\mathcal{M}\sigma}$  by applying the relational dependencies as specified in the model to the specific instances of the relational skeleton. Figure 3 shows the ground graph for the model of Figure 1 applied on the relational skeleton of Figure 2. In this work, we restrict our attention to relational models that do not contain the kind of relational autocorrelation that gives rise to cycles in the ground graph.

### 3 TEMPORAL RELATIONAL MODELS

Relational models can be extended with a temporal dimension to model probabilistic dependencies over time. Such an extension is similar to the way in which dynamic Bayesian networks (DBNs) extend Bayesian networks [Murphy, 2002]. A temporal relational model can be thought of as a sequence of time points, each of which is associated with a (non-temporal) relational model, and a set of dependencies that cross time points. Because dependencies in this model have a causal interpretation, dependencies across time points are only directed from the past to the future.

In this section, we extend the relational notions presented in Section 2 to include time. We assume that (1) time is discrete; (2) the schema is static; (3) relational dependencies do not change over time; (4) the temporal relational skeleton is given *a priori*; (5) the first-order Markov assumption

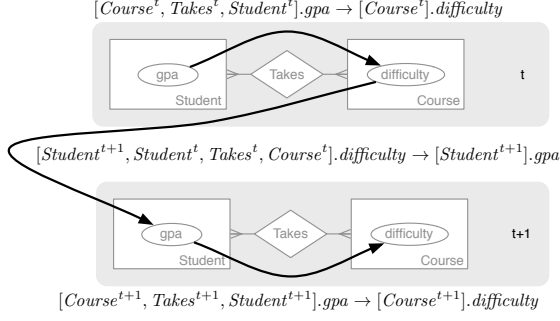


Figure 4: Example structure of a temporal relational model.

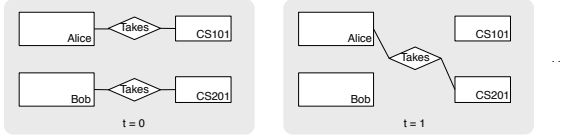


Figure 5: Example temporal relational skeleton for the schema shown in Figure 4.

holds (i.e., treatment and outcome can be at most one time point apart); and (6) all entities participating in a relationship are contemporaneous with the relationship.

Under these assumptions, the structure of a temporal relational model can be represented by using only two time points, as shown in Figure 4. Every time point has the same relational schema, shown in gray. A *temporal relational skeleton* provides a partial instantiation of a temporal relational schema. It specifies the entity and relationship instances that exist in each time point. Note that different sets of entity and relationship instances may be present in each time step. An example temporal relational skeleton is shown in Figure 5. In this case, both time points have the same set of entity instances (Alice and Bob are instances of the *Student* entity, CS101 and CS201 are instances of the *Course* entity). However, the instances of the relationship *Takes* differ. In  $t = 0$ , Alice takes CS101 and Bob takes CS201. In  $t = 1$ , Alice takes CS201 and Bob takes no classes.

*Temporal relational paths* capture possible ways to traverse the temporal schema; therefore, they can cross time points. For example, a temporal relational path is  $[Student^{t+1}, Student^t, Takes^t, Course^t]$ , which describes the classes that a student took in the previous semester. More formally, a temporal relational path is a sequence of non-temporal relational paths (relational paths within a time point) and “jumps” between neighboring time points. These jumps can happen at both entities and relationships because each choice encodes a distinct semantics. For example, the relational path  $[Student^{t+1}, Student^t, Takes^t, Course^t]$  describes the courses that a student took in the previous semester, while the path  $[Student^{t+1}, Takes^{t+1}, Course^{t+1}, Course^t]$

first finds the courses that a student is taking this semester, and then finds those courses in the previous semester. In the example skeleton of Figure 5, for Alice, the first path would reach CS101 at  $t = 0$ , while the second path, will reach CS201 at  $t = 0$ .

**Definition 1.** A *temporal relational path*  $P$  is a sequence of non-temporal relational paths  $P_0^{t_0}, \dots, P_k^{t_k}$  ( $k \geq 0$ ) such that for any two consecutive paths  $P_i^{t_i}, P_j^{t_j}$  in  $P$  the following hold:

1.  $|t_i - t_j| = 1$
2. The last item class of  $P_i^{t_i}$  is the same as the first item class of  $P_j^{t_j}$ .
3. No subpath of  $P$  is of the form  $[I_k^t, \dots, I_k^t]$ , where all relations in the subpath are one-to-one.

We use the notation  $time(P)$  to denote the set of all time points that appear in path  $P$ .

*Temporal relational variables* consist of a temporal relational path and an attribute that can be reached through that path. *Temporal relational dependencies* define probabilistic dependencies between two temporal relational variables. Temporal probabilistic dependencies are never directed backwards in time. Therefore, at the model level, there are no dependencies going back in time. However, the temporal constraints associated with a dependency are also implicitly encoded by the temporal relational path that annotates the dependency. To account for this, we forbid the temporal relational path of the treatment to go through any time points later than the time point of the outcome.

**Definition 2.** A *temporal relational dependency* consists of two temporal relational variables with a common base item,  $[I_1^t, \dots, I_k^t].V_k \rightarrow [I_1^t].V_1$  such that

$$\max \left( time([I_1^t, \dots, I_k^t]) \right) \leq t$$

The first-order Markov assumption for temporal causal models implies that for every probabilistic dependency, if the treatment is in time point  $t$ , then the outcome is either in  $t$  or in  $t + 1$ . In the case of relational domains, the relational path of the treatment carries some temporal information since it can contain multiple time points. For the first-order Markov condition to hold, we require the relational path of the treatment to only go through the current and the next time points. More formally, a temporal relational dependency  $[I_1^t, \dots, I_k^t].V_k \rightarrow [I_1^t].V_1$  follows the first-order Markov assumption if the following two conditions hold:

$$t = t' \text{ or } t = t' + 1 \quad (1)$$

$$time([I_1^t, \dots, I_k^t]) \subseteq \{t, t - 1\} \quad (2)$$

The structure of a 2-slice temporal relational model is defined as a set of temporal relational dependencies over a relational schema.



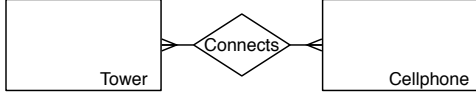


Figure 6: Relational schema for the reality mining dataset.

**Definition 3.** The structure of a 2-slice temporal relational model is a pair  $\mathcal{M} = \langle \mathcal{S}, \mathcal{D}_T \rangle$ , where  $\mathcal{S}$  is a relational schema and  $\mathcal{D}_T$  is a set of temporal relational dependencies that adhere to the first-order Markov assumption.

Figure 4 shows the structure of a 2-slice temporal relational model for the university domain. The temporal dependency shows that the difficulty of a course in the fall semester affects the spring GPA of the students that took this class in the fall. If, for example, a student takes many difficult classes, it is more likely that the student’s GPA will drop in the next semester. Finally, given a temporal relational skeleton  $\sigma_T$  and a temporal relational model  $\mathcal{M}$ , we can construct a temporal ground graph  $GG_{\mathcal{M}\sigma_T}$  in the same way as in the non-temporal case.

## 4 EXPRESSIVENESS OF TEMPORAL RELATIONAL MODELS

The temporal relational model described so far subsumes propositional directed networks (Bayesian networks), propositional temporal directed networks (dynamic Bayesian networks), and relational non-temporal models. This added expressivity comes at the cost of increased complexity. This raises an obvious and important question: What is the value of this added expressivity?

As an example of the practical utility of this added expressivity, we consider a real dataset and show how a path with temporal jumps leads to different terminal sets. Specifically, we used the Reality Mining dataset [Eagle and Pentland, 2006]. The dataset contains two entities, *Tower* and *Cellphone*, and one relationship, *Connects*. The relational schema for this domain is shown in Figure 6. For this dataset, entity instances (i.e., the set of towers and cellphones) do not change over time. However, the set of relationship instances (the connections) are different at every time point. In total, there are 95 cellphones, 32,579 towers, and 3,308,709 connections. Every connection is time-stamped with precision of 1 second. For our work, the time granularity was coarsened to a day.

We computed the terminal sets for the following three paths:

- $P1 : [Tower^{t+1}, Tower^t, Connects^t, Cellphone^t]$
- $P2 : [Tower^{t+1}, Connects^{t+1}, Connects^t, Cellphone^t]$
- $P3 : [Tower^{t+1}, Connects^{t+1}, Cellphone^{t+1}, Cellphone^t]$

The first path corresponds to the cellphones that were con-

Table 1: Jaccard distance between the terminal sets of the different paths for the reality mining dataset (100 sample dates, distance is averaged across dates and across towers).

| Jaccard distance | P1 vs. P3 | P1 vs. P2 | P2 vs. P3 |
|------------------|-----------|-----------|-----------|
| mean             | 0.47      | 0.31      | 0.31      |
| min              | 0         | 0         | 0         |
| max              | 1         | 1         | 1         |
| median           | 0.5       | 0         | 0         |

nected to a tower in the previous timestep. The second path corresponds to the cellphones that connected to a tower both in the current and in the previous timestep. The third path corresponds to the cellphones that connected to a tower in the current time step, and gets the state of those cellphones in the previous timestep.

We randomly selected 100 dates from the dataset. For each of these dates and for each tower that was used in these dates, we computed the terminal sets for the above paths. For a given tower and date, we computed the Jaccard distance between the different terminal sets. The Jaccard distance between two sets  $A$  and  $B$  is defined as  $J(A, B) = 1 - \frac{|A \cap B|}{|A \cup B|}$ . Intuitively, this quantifies the overlap of two sets, while accounting for the size of both. Table 1 shows the average Jaccard distance between the terminal sets, averaged across the dates and the towers. The results indicate that, on average, the terminal sets reached through the three paths will be different; therefore, this more expressive representation could be of use in real data.

## 5 TEMPORAL ABSTRACT GROUND GRAPHS

An abstract ground graph is a lifted representation that abstracts paths of dependence over all possible ground graphs for a given relational model [Maier *et al.*, 2013b]. Abstract ground graphs are shown to be sound and complete in the sense that every edge in the abstract ground graph corresponds to an edge in some ground graph, and every edge in an arbitrary ground graph is represented by an edge in an abstract ground graph. In this section we adapt the definition of abstract ground graphs for the case of a 2-slice temporal relational model.

**Definition 4.** A temporal abstract ground graph  $tAGG_{\mathcal{M}Bh} = \langle V, E \rangle$  for a 2-slice temporal relational model  $\mathcal{M} = \langle \mathcal{S}, \mathcal{D}_T \rangle$ , perspective  $B \in \mathcal{E} \cup \mathcal{R}$ , and hop threshold  $h \in \mathbb{N}^0$  is an abstraction of the dependencies  $\mathcal{D}_T$  for all possible ground graphs  $GG_{\mathcal{M}\sigma_T}$  of  $\mathcal{M}$  on arbitrary temporal skeletons  $\sigma_T$ . The temporal abstract ground graph is a directed graph with the following nodes and edges:

1.  $V = RV \cup IV$ , where

(a)  $RV$  is the set of *temporal relational variables* with a path of length at most  $h + 1$ .

$$RV = \{[B^t, \dots, I_j^t].V \mid \text{length}([B^t, \dots, I_j^t]) \leq h + 1\}$$

(b)  $IV$  are *intersection variables* between pairs of temporal relational variables that could intersect<sup>1</sup>.

$$IV = \{X \cap Y \mid X, Y \in RV\}$$

$$\text{and } X = [B^t, \dots, I_k^t, \dots, I_j^t].V$$

$$\text{and } Y = [B^t, \dots, I_l^t, \dots, I_j^t].V \text{ and } I_k^t \neq I_l^t\}$$

2.  $E = RVE \cup IVE$ , where

(a)  $RVE \subset RV \times RV$  are the *relational variable edges*:

$$RVE = \{[B^t, \dots, I_k^t].V_k \rightarrow [B^t, \dots, I_j^t].V_j \mid \\ [I_j^t, \dots, I_k^t].V_k \rightarrow [I_j^t].V_j \in \mathcal{D}_T \text{ and} \\ [B^t, \dots, I_k^t] \in \text{extend}([B^t, \dots, I_j^t], [I_j^t, \dots, I_k^t])\}$$

(b)  $IVE \subset (IV \times RV) \cup (RV \times IV)$  are the *intersection variable edges*. This is the set of edges that intersection variables “inherit” from the relational variables that they were created from.

The *extend* method converts dependencies of the model, specified in the canonical form, into dependencies from the perspective of the abstract ground graph.

Temporal abstract ground graphs can be shown to be a correct abstraction over all possible temporal ground graphs. The proof follows the one provided for the non-temporal abstract ground graphs, as presented by Maier *et al.* [2013b].

The temporal abstract ground graph for a model on the student-courses domain with the dependency

$$[Student^{t+1}, Student^t, Takes^t, Course^t].difficulty \rightarrow [Student^{t+1}].gpa$$

is shown in Figure 7. This abstract ground graph is from the perspective of *Student* and for hop threshold  $h = 4$ . Disconnected nodes are omitted.

## 6 $d$ -SEPARATION IN TEMPORAL ABSTRACT GROUND GRAPHS

The rules of  $d$ -separation provide a graphical criterion that specifies whether two sets of variables in a directed acyclic graph are conditionally independent given a third set of variables. Specifically, let  $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$  be disjoint sets of variables in a directed acyclic graph.  $\mathbf{X}$  is  $d$ -separated from  $\mathbf{Y}$

<sup>1</sup>Only relational variables in the same time point can intersect.

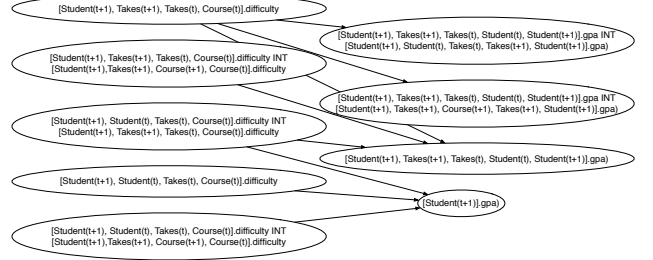


Figure 7: Temporal abstract ground graph from the perspective of *Student* and hop threshold  $h = 4$  for a model with one dependency:  $[Student^{t+1}, Student^t, Takes^t, Course^t].difficulty \rightarrow [Student^{t+1}].gpa$ . INT denotes intersection variables between pairs of temporal relational variables.

given  $\mathbf{Z}$  if every path between variables in  $\mathbf{X}$  and  $\mathbf{Y}$  is not a  $d$ -connecting path given  $\mathbf{Z}$ . A path is  $d$ -connecting given  $\mathbf{Z}$  if for every collider  $W$  on the path, either  $W \in \mathbf{Z}$  or a descendant of  $W$  is in  $\mathbf{Z}$ , and every non-collider on the path is not in  $\mathbf{Z}$ . Geiger and Pearl [1988] and Verma and Pearl [1988] showed that  $d$ -separation is sound and complete.

Ground graphs (and temporal ground graphs) are directed acyclic graphs; therefore, the rules of  $d$ -separation can be applied to them. In the case of domains where the first-order Markov assumption holds, a  $d$ -separating set for variables in the same time point can be found by examining only one time point in the past.

**Proposition 5.** Let  $G$  be a temporal directed acyclic graph that follows the first-order Markov assumption. (i) If  $X^{t+1}$  and  $Y^{t+1}$  are conditionally independent given some set  $\mathbf{Z}$ , then there exists a separating set  $\mathbf{W}$  such that  $\text{time}(\mathbf{W}) \subseteq \{t, t + 1\}$ . (ii) Similarly, if  $X^t$  and  $Y^{t+1}$  are conditionally independent given some set  $\mathbf{Z}$ , then there exists a separating set  $\mathbf{W}$  such that  $\text{time}(\mathbf{W}) \subseteq \{t, t + 1\}$ .

*Proof.* For each case, we will construct an appropriate separating set:

(i) Let  $\mathbf{W} = \text{parents}(X^{t+1}) \cup \text{parents}(Y^{t+1})$ . Because of the first-order Markov assumption,  $\text{time}(\mathbf{W}) \subseteq \{t, t + 1\}$ . Moreover, conditioning of  $\mathbf{W}$  renders  $X^{t+1}, Y^{t+1}$  independent because of the local Markov property (either  $X^{t+1}$  is a descendant of  $Y^{t+1}$  or vice versa, or none of them is a descendant of the either).

(ii) In this case, let  $\mathbf{W} = \text{parents}(Y^{t+1})$ . Because of the temporal semantics,  $X^t$  is a non-descendant of  $Y^{t+1}$ ; therefore,  $Y^{t+1}$  is conditionally independent of its non-descendants (that include  $X^t$ ) given  $\mathbf{W}$ .  $\square$

The significance of the above proposition is that, given a model where the first-order Markov assumption holds, we

could infer conditional independencies (and use them to learn the structure of the model) by only considering consecutive time points.

Maier *et al.* [2013b] showed that  $d$ -separation cannot be applied directly to a relational model. To correct for that, they introduced *relational  $d$ -separation*, a graphical criterion that can be applied to abstract ground graphs and used to infer conditional independencies that hold across all possible ground graphs of the model. Here, we show that the notion of relational  $d$ -separation can be generalized for temporal abstract ground graphs as well. In the following definition,  $X|_b$  denotes the terminal set of  $X$  starting at  $b$ , i.e., the set of  $X$  instances that can be reached if we start from instance  $b$  and follow the relational path of  $X$  on the relational skeleton.

**Definition 6** (Temporal relational  $d$ -separation). Let  $\mathbf{X}$ ,  $\mathbf{Y}$ , and  $\mathbf{Z}$  be three disjoint sets of temporal relational variables from perspective  $B$  for a 2-slice temporal relational model  $\mathcal{M}$  such that not both  $\mathbf{X}$  and  $\mathbf{Y}$  contain variables in  $t$ . Then  $\mathbf{X}$  and  $\mathbf{Y}$  are  $d$ -separated by  $\mathbf{Z}$  if and only if, for any temporal skeleton  $\sigma_T$ ,  $\mathbf{X}|_b$  and  $\mathbf{Y}|_b$  are  $d$ -separated by  $\mathbf{Z}|_b$  in ground graph  $GG_{\mathcal{M}\sigma_T}$  for all  $b \in \sigma_T(B)$ .

The following theorem shows that temporal relational  $d$ -separation is sound and complete up to a specified hop threshold. We use the notation  $\bar{\mathbf{X}}$  to denote the set of relational variables  $\mathbf{X}$  augmented with the set of intersection variables they participate in.

**Theorem 7.** Let  $\mathbf{X}$ ,  $\mathbf{Y}$ , and  $\mathbf{Z}$  be three disjoint sets of temporal relational variables for perspective  $B$  such that not both  $\mathbf{X}$  and  $\mathbf{Y}$  contain variables in  $t$ . Then, for any temporal skeleton  $\sigma_T$  and for all  $b \in \sigma(B)$ ,  $\mathbf{X}|_b$  and  $\mathbf{Y}|_b$  are  $d$ -separated by  $\mathbf{Z}|_b$  up to  $h$  in ground graph  $GG_{\mathcal{M}\sigma_T}$  if and only if  $\bar{\mathbf{X}}$  and  $\bar{\mathbf{Y}}$  are  $d$ -separated by  $\bar{\mathbf{Z}}$  on the abstract ground graph  $tAGG_{\mathcal{M}Bh}$ .

*Proof.* We prove this by defining a non-temporal relational model that is equivalent to the given temporal model. Since  $d$ -separation is sound and complete for non-temporal relational models, the result extends to the equivalent model, and therefore, the temporal relational model. Given a 2-slice temporal relational model  $\mathcal{M}_T = \langle \mathcal{S}, \mathcal{D}_T \rangle$ , construct a relational model  $\mathcal{M} = \langle \mathcal{S}', \mathcal{D} \rangle$  as follows:

- For every item in  $\mathcal{S}$ , add an item in  $\mathcal{S}'$  with superscript  $t$  and one with superscript  $t + 1$ :  $\mathcal{S}' = \{I^t, I^{t+1} \mid I \in \mathcal{S}\}$ .
- For every entity  $E \in \mathcal{S}$ , add in  $\mathcal{S}'$  a 1-1 relation between  $E^t$  and  $E^{t+1}$ .
- The set of relational dependencies is the set of temporal relational dependencies:  $\mathcal{D} = \mathcal{D}_T$ .

It can be shown that these two models are equivalent in the sense that there is a one-to-one correspondence between valid paths in  $\mathcal{M}_T$  and  $\mathcal{M}$ . Leveraging the temporal constraints of  $d$ -separation described by Proposition 5

and the soundness and completeness of  $d$ -separation for abstract ground graphs, we conclude that  $d$ -separation is sound and complete (up to a hop threshold) in temporal abstract ground graphs.  $\square$

## 7 TEMPORAL RCD

The theory of temporal relational  $d$ -separation allows us to derive all conditional independence facts that are consistent with the structure of a temporal relational model, the same way that  $d$ -separation connects the structure of a Bayesian network and the conditional independencies of the underlying distribution. This is precisely the connection that constraint-based algorithms leverage in order to learn the structure of models. Thus, temporal relational  $d$ -separation (and temporal abstract ground graphs) enable a constraint-based algorithm, TRCD<sup>2</sup>, that learns the structure of temporal and relational causal models from data.

TRCD extends RCD [Maier *et al.*, 2013a] to operate over a 2-slice temporal relational model. More specifically, it constructs a set of temporal abstract ground graphs, one from the perspective of each entity, and uses the theory of temporal relational  $d$ -separation on temporal abstract ground graphs to decide conditional independence facts. TRCD uses the temporal abstract ground graphs to determine which conditional independence facts should be checked in the data and which dependencies (edges) in the temporal relational model are implied by those facts. As in the case of RCD, for practical reasons, the space of potential dependencies is limited by a domain-specific hop threshold.

TRCD operates in two phases. Phase I learns a set of undirected dependencies and Phase II employs a set of orientation rules to orient those dependencies. Phase II of TRCD uses the same orientation rules as RCD—collider detection, known non-colliders (KNC), cycle avoidance (CA), Meek rule 3 (MR3), and relational bivariate orientation (RBO). Additionally, TRCD orients dependencies that cross time points from the past to the future.

RCD was shown to be sound and complete in the sample limit (i.e., with perfect tests of conditional independence) and for infinite hop threshold, under the standard assumptions of the causal Markov condition, faithfulness, and causal sufficiency for relational domains. By leveraging the soundness and completeness of temporal relational  $d$ -separation, TRCD can be shown to be sound and complete (under the same assumptions).

<sup>2</sup>Code available at [kdl.cs.umass.edu/trcd](http://kdl.cs.umass.edu/trcd).

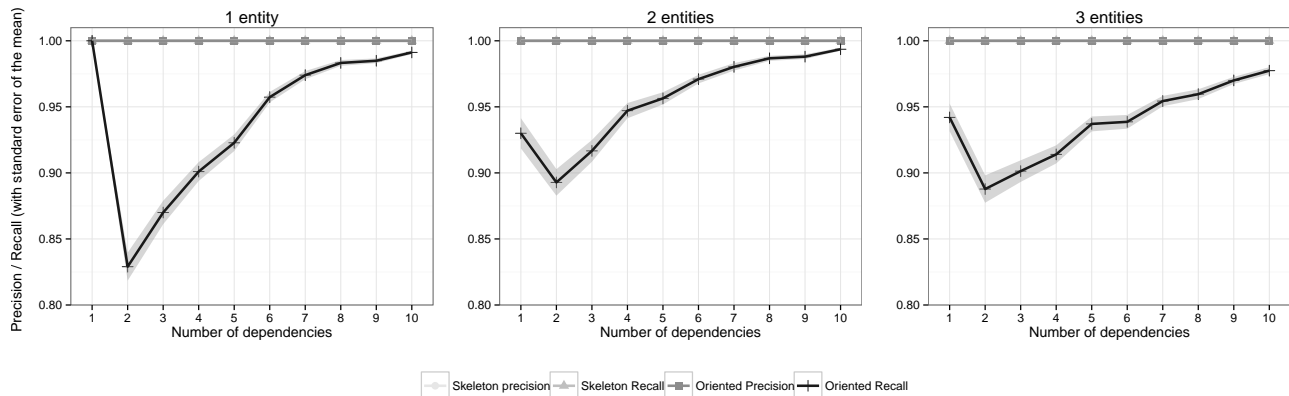


Figure 8: Precision and recall for TRCD after Phase I (unoriented) and after Phase II (oriented) when using an oracle for answering  $d$ -separation queries. The y-axis values start at 0.8.

## 8 EXPERIMENTS

### 8.1 ORACLE EXPERIMENTS

The goal of this set of experiments is twofold. First, we evaluate the theoretical performance of TRCD in the large sample limit. Towards that end, we employ an oracle for answering  $d$ -separation queries in the place of statistical tests of independence. Second, these experiments provide experimental evidence about the correctness of temporal relational  $d$ -separation.

We generated synthetic schemas with number of entities varying from 1 to 3, number of relationships fixed to one less than the number of entities, and number of attributes for each item drawn from  $Poisson(\lambda = 1) + 1$ . The cardinalities were uniformly selected at random. For each number of entities specified, we generated 500 different schemas. This generating process yielded 1,500 different schemas. For a given schema, we generated 10 different models with number of dependencies ranging from 1 to 10. Specifically, we generated all possible dependencies, up to hop-threshold  $h = 3$ . Then, we chose the desired number of dependencies from that space at random, subject to the following constraints: Each relational variable has at most 3 parents and the generated model contains no cycles. Moreover, every model must contain at least one dependency with a temporal relational path that contains more than one time point. This procedure resulted in a total of 15,000 models.

We ran TRCD with a  $d$ -separation oracle for each model. Figure 8 shows average precision and recall after Phase I (unoriented dependencies) and after Phase II (partially oriented dependencies). As expected, given that these experiments use an oracle, the algorithm always learns the correct set of unoriented dependencies (unoriented precision and recall are always 1). The algorithm makes no mistakes in the orientation (oriented precision is always 1); however,

Table 2: Frequency of the most-used orientation rules during Phase II of TRCD for the oracle experiments. For the rest of the rules, the frequency was less than 1%. Temporal dependencies are not oriented by the orientation rules.

| Number of entities | Collider detection | KNC | RBO | Percent of temporal dependencies |
|--------------------|--------------------|-----|-----|----------------------------------|
| 1                  | 71%                | 28% | 0%  | 66%                              |
| 2                  | 66%                | 11% | 23% | 68%                              |
| 3                  | 53%                | 11% | 36% | 65%                              |

it is not possible to orient all dependencies (oriented recall is lower than 1). Note that comparing to an oracle version of RCD is not straightforward. The oracle requires a true model that is fully directed. Converting the true temporal model to a non-temporal one would often result in cycles or undirected edges, and the relational  $d$ -separation oracle cannot be used.

Table 2 shows how often each orientation rule was used in Phase II. Collider detection, known non-colliders (KNC), and relational bivariate orientation (RBO) are orienting the majority of the edges. As expected, in the case of propositional models (one entity), the relational bivariate orientation rule, a rule for edge orientation that is unique to relational data, is never used. We observe that the other two rules—cycle avoidance and Meek’s rule 3—do not fire often. That can be explained by the fact that those rules would never fire in the presence of a temporal edge. Consider cycle avoidance in the propositional case: If the pattern  $X \rightarrow Y \rightarrow Z$  and  $X - Y$  is encountered, then cycle avoidance orients  $X \rightarrow Y$ . If any of the dependencies  $X \rightarrow Y \rightarrow Z$  crosses time points, then  $X$  and  $Y$  would be in different time points and the edge between them would be oriented based on temporal precedence. A similar argument can be made for the case of Meek’s rule 3.

## 8.2 EXPERIMENTS ON SYNTHETIC DATA

This experiment showcases the use of TRCD on data, i.e., without the use of an oracle to decide conditional independence. Towards that end, we generated synthetic models, generated data from these models, and applied TRCD on them. The use of synthetic data allows us to have access to the ground truth, so we can measure the accuracy of the learned models. The data-generating process is described in detail below.

Using the same process as described in 8.1, we generated 5 synthetic schemas with 2 entities and 5 synthetic schemas with 3 entities. For each schema, we generated 10 models with number of dependencies ranging from 1 to 10. This resulted in 100 different models. For each model we created 3 different relational skeletons over 300 timepoints. The number of entity instances at each time point was drawn from  $Poisson(\lambda) + 1$ , where  $\lambda \sim \mathcal{U}(5, 10)$ . The degree distribution for the relationship instances was drawn from  $Poisson(\lambda) + 1$ , where  $\lambda \sim \mathcal{U}(1, 5)$ . Regarding the parameters of the graphical model, the marginals were parameterized as normal distributions  $\mathcal{N}(\mu, \sigma)$ , where  $\mu \sim \mathcal{U}(0, 5)$  and  $\sigma \sim \mathcal{U}(0, 1)$ . The conditional distribution for a relational variable  $X$  was  $\sum_{Y \in \text{parents}(X)} (\text{avg}(Y)) + 0.1 * \mathcal{N}(0, 1)$ . This resulted in 300 datasets, each over 300 time points.

In order to assess statistical independence, we fitted a standard linear least-squares regression equation to the outcome variable using the treatment and the variables in the conditioning set as covariates. For relational variables in the conditioning set, we used the average as the aggregation function. Then, we directly used the t-test of the coefficient of the treatment variable to assess independence ( $p > 0.05$  or effect size  $< 0.01$ ). Figure 9 shows average precision and recall of TRCD after Phase I and Phase II, when applied to the synthetic datasets. While precision after Phase I is more than 0.75 in most cases, the recall after Phase I is relatively low. That implies that we concluded independence (and therefore we removed an edge) more often than we should. This corresponds to Type II errors and can be attributed to the lack of good conditional independence tests for relational data.

Finally, to demonstrate the difference in expressiveness between TRCD and RCD (the only constraint-based algorithm for relational data), we ran RCD on a “temporally flattened”<sup>3</sup> version of the synthetic data. The true model and the model that TRCD learned are shown in Figure 10. The model learned by RCD is shown in Figure 11. TRCD correctly learns and orients three of the edges, with the correct path specification. Those dependencies cannot even be expressed in the space of dependencies for RCD.

<sup>3</sup>RCD is ignoring temporal information. An instance is uniquely identified by instance id and time point.

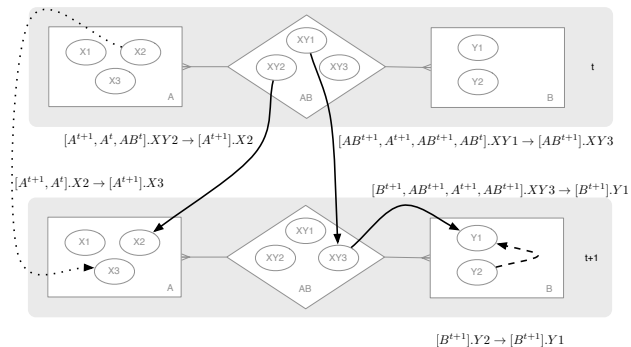


Figure 10: True temporal relational model. The dotted edge was not learned by TRCD, while the dashed edge was left unoriented.

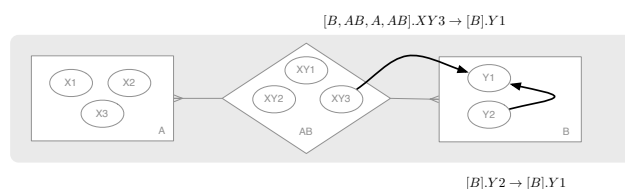


Figure 11: Model learned by RCD for data generated from the temporal model of Figure 10.

## 9 RELATED WORK

For the propositional case, there are several approaches for learning the structure of temporal probabilistic models. Most of them are not constraint-based methods, but follow the search-and-score paradigm. Friedman *et al.* [1998] present an algorithm to learn the structure of DBNs from complete data using a search-and-score algorithm, and from incomplete data using structural EM. Lähdesmäki and Shmulevich [2008] use Bayesian methods to learn the structure of a DBN from steady state measurements or from time-series data and steady state measurements. Robinson and Hartemink [2010] present an MCMC algorithm to learn the structure of a DBN that changes over time. A different approach that learns a causal temporal model from time series data is the difference-based causal learner [Voortman *et al.*, 2010]. This framework is based on dynamic Structural Equation Models.

A constraint-based method for temporal propositional domains is presented by Entner and Hoyer [2010] who extend the FCI algorithm [Spirtes *et al.*, 2000] to temporal domains. FCI relaxes the causal sufficiency assumption, i.e., it allows the presence of latent common causes. Our approach is the first approach that uses a constraint-based algorithm for data that is both temporal *and* relational (although under the assumption of causal sufficiency).

Another widely used method for inferring causal relationships from temporal data is Granger causality [Granger, 1969]. The main idea underlying Granger causality is that

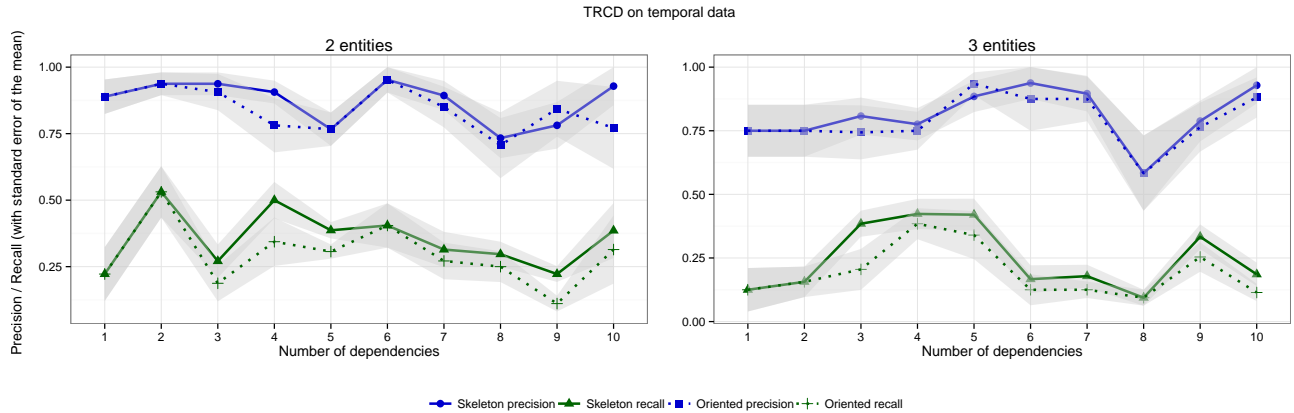


Figure 9: TRCD on synthetic temporal data.

a cause should improve the predictive accuracy of its effect, compared to predictions based solely on the effect’s past values. There has been work in extending Granger causality for multivariate settings, as well as in combining Granger causality with graphical models [Eichler, 2012]. Liu *et al.* [2010] propose a regularized Hidden Markov Random Field regression to learn the structure of a temporal causal graph from multivariate time-series data. However, the methods used for learning the structure of the causal model are not constraint-based.

Another line of work in learning temporal and relational models stems from combining first-order logic with probabilistic frameworks. Logical Hidden Markov Models extend Hidden Markov Models to handle relational (non-flat data) [Kersting *et al.*, 2006]. Kersting and Raiko [2005] provide an EM-based algorithm to learn the structure of LHMMs.

In terms of representation, Manfredotti [2009] introduces relational dynamic Bayesian networks (RDBNs), a first-order logic-based extension of dynamic Bayesian networks. RDBNs are similar to the relational model we define; however, we provide an explicit characterization for the space of relational paths, and subsequently, the space of relational dependencies. To be more specific, temporal relational paths in our framework are restricted to conjunctions of predicates that correspond to possible traversals of the relational schema. This restriction, together with the domain specific hop threshold, allows us to enumerate the space of potential dependencies to learn.

## 10 CONCLUSIONS AND FUTURE WORK

In this paper we presented a formalization of temporal relational models, and we extended the theory of relational  $d$ -separation to the temporal domain. We presented a constraint-based algorithm, TRCD, that leverages the notion of temporal relational  $d$ -separation to learn the causal

structure of temporal relational models from data. We showed that the algorithm is sound and complete, and we provided experimental evidence that showcases the correctness of TRCD. Finally, we showed the improvement that TRCD achieves compared to RCD when applied to domains with a temporal component.

TRCD makes certain simplifying assumptions. Future work could focus on relaxing some of those assumptions, specifically, allowing the structure of the causal model to change over time (change point detection for relational data). Another avenue for future research is relaxing the causal sufficiency assumption by employing techniques such as blocking [Rattigan *et al.*, 2011] for temporal relational domains. Finally, an important issue that arises when modelling time as a discrete quantity is how to choose the appropriate granularity of time points. Ribeiro *et al.* [2012] provide an analysis on how aggregating at a given time granularity affects the characterization of the underlying temporal process. Continuous time Bayesian networks [Nodelman *et al.*, 2002, 2003] provide a way around this by allowing each variable to be modeled at a different time granularity.

## Acknowledgements

Funding was provided by the U.S. Army Research Office (ARO) and Defense Advanced Research Projects Agency (DARPA) under Contract Number W911NF-11-C-0088. The content of the information in this document does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

## References

- Nathan Eagle and Alex (Sandy) Pentland. Reality mining: Sensing complex social systems. *Personal and Ubiquitous Computing*, 10(4):255–268, March 2006.
- Michael Eichler. Graphical modelling of multivariate time series. *Probability Theory and Related Fields*, 153(1-2):233–268, 2012.
- Doris Entner and Patrik Hoyer. On causal discovery from time series data using FCI. In *Proceedings of the 5th European Workshop on Probabilistic Graphical Models (PGM-2010)*, pages 121–128, 2010.
- Nir Friedman, Kevin Murphy, and Stuart Russell. Learning the structure of dynamic probabilistic networks. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, pages 139–147, 1998.
- Dan Geiger and Judea Pearl. On the logic of causal models. In *Proceedings of the Fourth Annual Conference on Uncertainty in Artificial Intelligence*, pages 136–147, 1988.
- Clive W. J. Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica*, 37(3):424–38, July 1969.
- Kristian Kersting and Tapani Raiko. “Say EM” for selecting probabilistic models for logical sequences. In *Proceedings of the Twenty-First Conference in Uncertainty in Artificial Intelligence*, pages 300–307, 2005.
- Kristian Kersting, Luc De Raedt, and Tapani Raiko. Logical hidden Markov models. *Journal of Artificial Intelligence Research*, 25:425–456, 2006.
- Harri Lähdesmäki and Ilya Shmulevich. Learning the structure of dynamic Bayesian networks from time series and steady state measurements. *Machine Learning*, 71(2-3):185–217, June 2008.
- Yan Liu, Alexandru Niculescu-Mizil, Aurelie C. Lozano, and Yong Lu. Learning temporal causal graphs for relational time-series analysis. In *Proceedings of the Twenty-Seventh International Conference on Machine Learning*, pages 687–694, 2010.
- Marc Maier, Katerina Marazopoulou, David Arbour, and David Jensen. A sound and complete algorithm for learning causal models from relational data. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, pages 371–380, 2013.
- Marc Maier, Katerina Marazopoulou, and David Jensen. Reasoning about Independence in Probabilistic Models of Relational Data. *arXiv:1302.4381*, 2013.
- Cristina E. Manfredotti. *Modeling and Inference with Relational Dynamic Bayesian Networks*. PhD thesis, University of Milano, 2009.
- Kevin P. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, University of California, Berkeley, 2002.
- Uri Nodelman, Christian R. Shelton, and Daphne Koller. Continuous time Bayesian networks. In *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence*, pages 378–387, 2002.
- Uri Nodelman, Christian R. Shelton, and Daphne Koller. Learning continuous time Bayesian networks. In *Proceedings of the Nineteenth International Conference on Uncertainty in Artificial Intelligence*, pages 451–458, 2003.
- Judea Pearl and Thomas Verma. A theory of inferred causation. In J. Allen, R. Fikes, and E. Sandewall, editors, *Principles of Knowledge Representation and Reasoning: Proceeding of the Second International Conference*, pages 441–452. Morgan Kaufmann, 1991.
- Matthew J.H. Rattigan, Marc Maier, and David Jensen. Relational blocking for causal discovery. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, pages 145–151, 2011.
- Bruno F. Ribeiro, Nicola Perra, and Andrea Baronchelli. Quantifying the effect of temporal resolution in time-varying network. *CoRR*, abs/1211.7052, 2012.
- Joshua W. Robinson and Alexander J. Hartemink. Learning non-stationary dynamic Bayesian networks. *Journal of Machine Learning Research*, 11:3647–3680, December 2010.
- Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction and Search*. MIT Press, Cambridge, MA, 2nd edition, 2000.
- Thomas Verma and Judea Pearl. Causal networks: Semantics and expressiveness. In *Proceedings of the Fourth Annual Conference on Uncertainty in Artificial Intelligence*, pages 352–359, 1988.
- Mark Voortman, Denver Dash, and Marek J. Druzdzel. Learning why things change: The difference-based causality learner. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, pages 641–650, 2010.

---

# Hamiltonian ABC

---

**Edward Meeds**  
Informatics Institute  
University of Amsterdam  
tmeeds@gmail.com

**Robert Leenders**  
Informatics Institute  
University of Amsterdam  
leenders.robert@gmail.com

**Max Welling\***  
Informatics Institute  
University of Amsterdam  
welling.max@gmail.com

## Abstract

Approximate Bayesian computation (ABC) is a powerful and elegant framework for performing inference in simulation-based models. However, due to the difficulty in scaling likelihood estimates, ABC remains useful for relatively low-dimensional problems. We introduce Hamiltonian ABC (HABC), a set of likelihood-free algorithms that apply recent advances in scaling Bayesian learning using Hamiltonian Monte Carlo (HMC) and stochastic gradients. We find that a small number forward simulations can effectively approximate the ABC gradient, allowing Hamiltonian dynamics to efficiently traverse parameter spaces. We also describe a new simple yet general approach of incorporating random seeds into the state of the Markov chain, further reducing the random walk behavior of HABC. We demonstrate HABC on several typical ABC problems, and show that HABC samples comparably to regular Bayesian inference using true gradients on a high-dimensional problem from machine learning.

## 1 INTRODUCTION

In simulation-based science, models are defined by a simulator and its parameters. These are called *likelihood-free* models because, in contrast to probabilistic models, their likelihoods are either intractable to compute or must be approximated by simulations. To perform inference in likelihood-free models, a broad class of algorithms called Approximate Bayesian Computation [3, 13, 20, 12] are employed.

At the core of every ABC algorithm is simulation. To evaluate the quality of a parameter vector  $\theta$ , a simulation is run

---

\*Donald Bren School of Information and Computer Sciences  
University of California, Irvine, and Canadian Institute for Advanced Research.

using  $\theta$  as inputs and producing outputs  $\mathbf{x}$ . If the pseudo-data  $\mathbf{x}$  is “close” to observations  $\mathbf{y}$ , then  $\theta$  is kept as a sample from the approximate posterior. Parameters  $\theta$  are then adjusted, depending upon the algorithm, to obtain the next sample.

In ABC, there is a fundamental trade-off between the computation required to obtain independent samples and the approximation to the true posterior. If the parameter measuring closeness is too small, then samplers “mix” poorly; on the other hand, if it is too large, then the approximation is poor. As the dimension of the parameters grows, the problem worsens, just as it does for general Bayesian inference with probabilistic models, but it is more acute for ABC due to its simulation requirement. There is therefore a deep interest in improving the efficiency of ABC samplers (in terms of computation per independent sample). In this paper we address this issue directly by using Hamiltonian dynamics to approximately sample from likelihood-free models with high-dimensional parameters.

Hamiltonian Monte Carlo (HMC) [7, 16] is perhaps the only Bayesian inference algorithm that scales to high-dimensional parameter spaces. The core computation of HMC is the gradient of the log-likelihood. Two problems arise if we consider HMC for ABC: one, how can the gradients be computed for high-dimensional likelihood-free models, and two, given a stochastic approximation to the gradient, can a valid HMC algorithm be derived?

To answer the latter, we turn to recent developments in scaling Bayesian inference using HMC and stochastic gradients [25, 5, 6]. We call these *stochastic gradient Hamiltonian dynamics* (SGHD) algorithms. SGHD algorithms are computationally efficient for two reasons. First, they avoid computing the gradient of the log-likelihood over the entire data set, instead approximating it using small batches of data, i.e. computing stochastic gradients. Second, they can maintain reasonable approximations to the Hamiltonian dynamics and therefore avoid a Metropolis-Hastings correction step involving the full data set. Different strategies are employed to do this: small step-sizes combined with Langevin dynamics [25] (stochastic gradi-



ent Langevin dynamics—SGLD), using friction to prevent accumulation of errors in the Hamiltonian [5] (stochastic gradient HMC—SGHMC), and using a thermostat to control the temperature of the Hamiltonian [6] (stochastic gradient Nose-Hoover thermostats—SGNHT). Each of these strategies can be used by HABC.

In HABC, we use forward simulations to approximate the likelihood-free gradient. The key difference between SGHD methods and HABC is that the stochasticity of the gradient does not come from approximating the full data gradient with a mini-batch gradient, but by the stochasticity of the simulator. It is therefore not the expense of the simulator (though this could very well be the case for many interesting simulation-based models – see Section 7) that requires an approximation to the gradient, but the likelihood-free nature of the problem.

There are several difficulties in estimating gradients of likelihood-free models that we address with HABC. The first is due to the form of the ABC log-likelihood. As we show in Section 2, using a conditional model for  $\pi(\mathbf{x}|\theta)$  provides an estimate of the ABC likelihood that is less sensitive to  $\epsilon$  and therefore is more conducive to stochastic gradient computations. The second difficulty is that for high-dimensional parameter spaces, computing the gradients naively (i.e. by finite differences (FD) [9]) can squash the gains brought by the Hamiltonian dynamics. Fortunately, we can use existing stochastic approximation algorithms [21, 22] that can be used to compute unbiased estimators of the gradient with a small number of forward simulations that is *independent* of the parameter dimension. The *stochastic perturbation stochastic approximation* (SPSA) [21] is described in Section 4

A further innovation of this paper is the use of persistent random numbers (PRNs) to improve the efficiency of the Hamiltonian dynamics. The idea behind PRNs is to use the same set of random seeds for estimating a gradient by FD or SPSA, i.e. when simulating  $\pi(\mathbf{x}|\theta + d\theta)$  and  $\pi(\mathbf{x}|\theta - d\theta)$  use the same random seeds. This was applied successfully to SPSA [10] (and is analogous to using the same mini-batch in stochastic gradient methods). We extend and simplify this approach by including the random seeds  $\omega$  into the state of the Markov chain; by keeping the random seeds fixed for several consecutive steps, the second order gradient stochasticity is greatly reduced. We show that doing this produces a valid MCMC procedure. This approach is not exclusive to HABC; our experiments show it also helps random-walk ABC-MCMC.

We briefly review ABC in Section 2. In Section 3 we review three approaches to stochastic gradient inference using Hamiltonian dynamics: SGLD, SGHMC, and SGNHT. We then introduce Hamiltonian ABC in Section 4, where we will show how to improve the stability of the gradient estimates by using PRNs and local density estimators

of the simulator. Extensions to high-dimensional parameter spaces are also discussed. In Section 5 we show how HABC behaves on a simple one-dimensional problem, then in Section 6 we compare HABC with ABC-MCMC for two problems: a low-dimensional model of chaotic population dynamics and a high-dimensional problem.

## 2 APPROXIMATE BAYESIAN COMPUTATION

Consider the Bayesian inference task of either drawing samples from or learning an approximate model of the following (usually intractable) posterior distribution:

$$\pi(\theta|\mathbf{y}_1, \dots, \mathbf{y}_N) \propto \pi(\theta)\pi(\mathbf{y}_1, \dots, \mathbf{y}_N|\theta) \quad (1)$$

where  $\pi(\theta)$  is a prior distribution over parameters  $\theta \in \mathbb{R}^D$  and  $\pi(\mathbf{y}_1, \dots, \mathbf{y}_N|\theta)$  is the likelihood of  $N$  data observations, where  $\mathbf{y}_i \in \mathbb{R}^J$ . In ABC, the vector of  $J$  observations are typically informative statistics of the raw observations. It can be shown that if the statistics used in the likelihood function are sufficient, then these algorithms sample correctly from an approximation to the true posterior [12]. The simulator is treated as a generator of random pseudo-observations, i.e.  $\mathbf{x} \stackrel{\text{sim}}{\sim} \pi(\mathbf{x}|\theta)$  is a draw from the simulator. Discrepancies between the simulator outputs  $\mathbf{x}$  and the observations  $\mathbf{y}$  are scaled by a closeness parameter  $\epsilon$  and treated as likelihoods. This is the equivalent to putting an  $\epsilon$ -kernel around the observations, and using a Monte Carlo estimate of the likelihood using  $S$  draws of  $\mathbf{x}$ :

$$\pi_\epsilon(\mathbf{y}|\theta) = \int \pi_\epsilon(\mathbf{y}|\mathbf{x})\pi(\mathbf{x}|\theta)d\mathbf{x} \approx \frac{1}{S} \sum_{s=1}^S \pi_\epsilon(\mathbf{y}|\mathbf{x}^{(s)}) \quad (2)$$

In ABC Markov chain Monte Carlo (MCMC) [13, 26] the Metropolis-Hastings (MH) proposal distribution is composed of the product of the proposal for the parameters  $\theta$  and the proposal for the simulator outputs:

$$q(\theta', \mathbf{x}^{(1)'}, \dots, \mathbf{x}^{(S)' }|\theta) = q(\theta'|\theta) \prod_s \pi(\mathbf{x}^{(s)'}|\theta') \quad (3)$$

Using this form of the proposal distribution, and using the Monte Carlo approximation eq 2, we arrive at the following Metropolis-Hastings accept-reject probability,

$$\alpha = \min \left( 1, \frac{\pi(\theta') \sum_{s=1}^S \pi_\epsilon(\mathbf{y}|\mathbf{x}^{(s)'})q(\theta|\theta')}{\pi(\theta) \sum_{s=1}^S \pi_\epsilon(\mathbf{y}|\mathbf{x}^{(s)})q(\theta'|\theta)} \right) \quad (4)$$

If the simulations are part of the Markov chain, the algorithm corresponds to the pseudo-marginal (PM) sampler [2], otherwise it is a marginal sampler [13, 20]. For this paper we will be interested in the PM sampler because this is equivalent to having the random states that generated the simulation outputs in the state of the Markov chain, which

we will use within a valid ABC sampling algorithm in Section 4.

An alternative approach to computing the ABC likelihood is to estimate the parameters of a conditional model  $\pi(\mathbf{x}|\boldsymbol{\theta})$ , e.g. using kernel density estimate [24] or a Gaussian model [28]. While either approach should be adequate and both have their own limits and advantages, for this paper we will use a Gaussian model. In ABC, using a conditional Gaussian model for  $\pi(\mathbf{x}|\boldsymbol{\theta})$  is called a *synthetic likelihood* (SL) model [28]. For a SL log-likelihood model, we compute estimators of the first and second moments of  $\pi(\mathbf{x}|\boldsymbol{\theta})$ . The advantage is that for a Gaussian  $\epsilon$ -kernel, we can convolve the two densities

$$\begin{aligned}\pi_{\epsilon}(\mathbf{y}|\boldsymbol{\theta}) &= \int \mathcal{N}(\mathbf{y}|\mathbf{x}, \epsilon^2)\mathcal{N}(\mathbf{x}|\mu_{\boldsymbol{\theta}}, \sigma_{\boldsymbol{\theta}}^2)d\mathbf{x} \quad (5) \\ &= \mathcal{N}(\mathbf{y}|\mu_{\boldsymbol{\theta}}, \sigma_{\boldsymbol{\theta}}^2 + \epsilon^2) \quad (6)\end{aligned}$$

Of particular concern to this paper is the behavior of the log-likelihoods for different values of  $\epsilon$ . In the  $\epsilon$ -kernel case, the log-likelihood is very sensitive to small values of  $\epsilon$ :

$$\begin{aligned}\log \pi_{\epsilon}(\mathbf{y}|\boldsymbol{\theta}) &= \log \sum_s \mathcal{N}(\mathbf{y}|\mathbf{x}^{(s)}, \epsilon^2) \quad (7) \\ &= \log \mathcal{N}(\mathbf{y}|\mathbf{x}^{(s)}, \epsilon^2) + \log(1 + H) \quad (8) \\ &\approx -\log \epsilon - \frac{1}{2\epsilon^2}(\mathbf{y} - \mathbf{x}^{(m)})^2 \quad (9)\end{aligned}$$

where  $m$  is the simulation that is closest to  $\mathbf{y}$ ,  $H$  is a sum over terms close to 0. We can see that the log-likelihood can be set arbitrarily small by decreasing  $\epsilon$ . On the other hand, by using a model of the simulation at  $\boldsymbol{\theta}$

$$\log \pi_{\epsilon}(\mathbf{y}|\boldsymbol{\theta}) \approx -\frac{1}{2} \log(\sigma_{\boldsymbol{\theta}}^2 + \epsilon^2) - \frac{(\mathbf{y} - \mu_{\boldsymbol{\theta}})^2}{2(\sigma_{\boldsymbol{\theta}}^2 + \epsilon^2)} \quad (10)$$

For the SL model,  $\epsilon$  acts as a smoothing term and can be set to small values with little change to the log-likelihood, as long as the SL estimators are fit appropriately. This insensitivity to  $\epsilon$  will be used in Section 4 for estimating gradients of the ABC likelihood. Before describing HABC in full detail however, we now explain how scaling Hamiltonian dynamics in Bayesian learning can be accomplished using stochastic gradients from batched data.

### 3 SCALING BAYESIAN INFERENCE USING HAMILTONIAN DYNAMICS

Scaling Bayesian inference algorithms to massive datasets is necessary for their continuing relevance in the so-called *big data* era. We now review the role stochastic gradient methods combined with Hamiltonian dynamics have played in recent advances in scaling Bayesian inference. Most importantly, these methods have combined the ability of HMC to explore high-dimensional parameter spaces

with the computational efficiency of using stochastic gradients based on small mini-batches of the full dataset. After an overview of HMC, we will briefly describe stochastic gradient Hamiltonian dynamics (SGHD), starting with using Langevin dynamics [25], then HMC with friction [5], and finally HMC with thermostats [6]. We will then make the connection between SGHD and HABC in Section 4.

#### 3.1 Hamiltonian Monte Carlo

Hamiltonian dynamics are often necessary to adequately explore the target distribution of high-dimensional parameter spaces. By proposing parameters that are far from the current location and yet have high acceptance probability, Hamiltonian Monte Carlo [7, 16] can efficiently avoid random walk behavior that can render proposals in high-dimensions painfully slow to mix.

HMC simulates the trajectory of a particle along a frictionless surface, using random initial momentum  $\boldsymbol{\rho}$  and position  $\boldsymbol{\theta}$ . The Hamiltonian function computes the energy of the system and the dynamics govern how the momentum and position change over time. The continuous Hamiltonian dynamics can be simulated by discretizing time into small steps  $\eta$ . If  $\eta$  is small, the value of  $\boldsymbol{\theta}$  at the end of a simulation can be used as proposals within the Metropolis-Hastings algorithm. Hamiltonian dynamics should propose  $\boldsymbol{\theta}$  that are always accepted, but errors due to discretization may require a Metropolis-Hastings correction. It is this correction step that SGHD algorithms want to avoid as it requires computing the log-likelihood over the full data set.

More formally, the Hamiltonian  $H(\boldsymbol{\theta}, \boldsymbol{\rho}) = U(\boldsymbol{\theta}) + K(\boldsymbol{\rho})$  is a function of the current potential energy  $U(\boldsymbol{\theta})$  and kinetic energy  $K(\boldsymbol{\rho}) = \boldsymbol{\rho}^T \mathbf{M}^{-1} \boldsymbol{\rho} / 2$  ( $\mathbf{M}$  is a diagonal matrix of masses which for presentation are set to 1). The potential energy is defined by the negative log joint density of the data and prior:

$$U(\boldsymbol{\theta}) = -\log \pi(\boldsymbol{\theta}) - \sum_{i=1}^N \log \pi(\mathbf{y}_i|\boldsymbol{\theta}) \quad (11)$$

The Hamiltonian dynamics follow

$$d\boldsymbol{\theta} = \boldsymbol{\rho} dt \quad d\boldsymbol{\rho} = -\nabla U(\boldsymbol{\theta}) dt \quad (12)$$

in simulation  $dt = \eta$ .

#### 3.2 Stochastic Gradient Hamiltonian Dynamics

If the log-likelihood over the full data set is replaced with a mini-batch estimate, as is done for the following *stochastic gradient Hamiltonian dynamics* (SGHD) algorithms, then the error in simulating the Hamiltonian dynamics comes not only from the discretization, but from the variance of the stochastic gradient. As long as this error is controlled, either by using small steps  $\eta$  (SGLD), or adding friction

terms  $B$  (SGHMC), or using a thermostat  $\xi$  (SGNHT), the expensive MH correction step can be avoided and values of  $\theta$  from the Hamiltonian dynamics can be used as approximate samples from the posterior. SGHD algorithms belong to a larger class of *noisy Monte Carlo* methods that target intractable likelihoods; see [1] for an extensive overview of noisy Monte Carlo.

We develop SGHD from the large-scale data case, where the intractability is due to computing the full potential energy and its gradient; it is approximated using mini-batches:

$$\begin{aligned}\hat{U}(\theta) &= -\log \pi(\theta) - \frac{N}{n} \sum_{i=h_1}^{h_n} \log \pi(\mathbf{y}_i|\theta) \quad (13) \\ \nabla \hat{U}(\theta) &= -\nabla \log \pi(\theta) - \frac{N}{n} \sum_{i=h_1}^{h_n} \nabla \log \pi(\mathbf{y}_i|\theta) \quad (14)\end{aligned}$$

where  $n$  is the mini-batch size, and  $h_i$  are indices chosen randomly without replacement from  $[1, N]$  (i.e. it defined a random mini-batch). In likelihood-free settings, the stochasticity of the potential energy due to the mini-batches is instead caused by simulation noise; further likelihood assumptions, such as a Gaussian model, add another layer of approximation to our posterior. Below we describe three SGHD algorithms, originally developed for large-scale data applications, but for which we will apply directly to likelihood-free inference using gradient approximations in Section 4.

**Stochastic gradient Langevin dynamics** (SGLD) [25] performs one full leap-frog step of HMC. In doing so, SGLD avoids explicitly computing updates for momenta  $\rho$ ; the update for  $\theta$  is

$$\theta_{t+1} = \theta_t + \eta \mathcal{N}(0, \mathbf{I}_p) - \eta^2 \nabla \hat{U}(\theta_t)/2 \quad (15)$$

One of the potential drawbacks of SGLD is that the momentum term is *refreshed* (implicitly) for every update of the  $\theta$ , and since this means the parameter update only uses the current gradient approximation, it limits the benefits of using Hamiltonian dynamics. On the other hand, this also prevents SGLD from accumulating errors in the Hamiltonian dynamics. SGLD has been applied to another intractable likelihood model, Gibbs random fields [1], which closely resembles how SGLD is applied in this paper.

**Stochastic Gradient HMC** (SGHMC) [5] avoids  $\rho$  refreshment altogether. SGHMC makes the assumption  $\nabla \hat{U}(\theta) = \nabla U(\theta) + \mathcal{N}(0, \mathbf{V}_\theta)$ , where  $\mathbf{V}_\theta$  is the covariance of the gradient approximation. To avoid a MH correction step at the end of a trajectory, a friction term  $\mathbf{B}$  proportional to  $\mathbf{V}_\theta$  is added to  $\Delta \rho$ . In practice, since we can only approximate  $\mathbf{B}$ , a user defined friction term  $\mathbf{C}$  is used. In our experiments we compute an online estimate  $\hat{\mathbf{V}}$  and set  $\mathbf{C} = c\mathbf{I}_p + \hat{\mathbf{V}}$ .

**Stochastic Gradient thermostats** (SGNHT) [6] addresses the difficulty of estimating  $\mathbf{B}$  by introducing a scalar variable  $\xi$  who's addition to the Hamiltonian dynamics maintains the temperature of the system constant, i.e. it acts as a (Nose-Hoover) thermostat [11].

## 4 HAMILTONIAN ABC

The general approach of applying Hamiltonian dynamics to ABC requires choosing one of the SGHD algorithms and then plugging in the ABC gradient approximation  $\nabla \hat{U}(\theta)$ . With this in mind we leave the details of the Hamiltonian updates to previous work [25, 5, 6] and focus on the details of how stochastic gradients are computed in the likelihood-free setting. Note that in our implementation, we do not use a MH correction (except when switching seeds), though this can easily be added for any particular problem.

### 4.1 Deterministic Representations of Simulations

Implicit in each simulation run  $\mathbf{x} \stackrel{\text{sim}}{\sim} \pi(\mathbf{x}|\theta)$  is a sequence of internally generated random numbers that are used to produce random draws from  $\pi(\mathbf{x}|\theta)$ . These random numbers are important to HABC because we wish to control the stochasticity of the simulator when computing its gradient. Furthermore, we will control the random numbers over multiple time steps. Instead of keeping track of random numbers, we can equivalently keep a vector of  $S$  random seeds  $\omega$ . This allows HABC to treat the simulation function  $\pi(\mathbf{x}|\theta)$  as a blackbox, outside of which we can control the random number generator (RNG), and represent  $\mathbf{x}^{(s)}$  as the output of a deterministic function; i.e.  $\mathbf{x}^{(s)} = f(\theta, \omega_s)$  instead of  $\mathbf{x}^{(s)} \stackrel{\text{sim}}{\sim} \pi(\mathbf{x}|\theta)$ . We include  $\omega$  as part of the state of our Markov chain.

### 4.2 Kernel- $\epsilon$ versus Synthetic-likelihood -based Gradients

In Section 2 we showed that the synthetic-likelihood representation of  $\mathcal{L}_\epsilon(\theta)$  is less sensitive to small choices of  $\epsilon$ . This is particularly important to HABC as our gradient approximations are proportional to differences in  $\mathcal{L}_\epsilon(\theta)$ ; if the variance of the stochastic gradients is too high, then we must choose a very small step-size  $\eta$ , eliminating the usefulness of HMC for ABC. Under the deterministic representation of  $\mathbf{x}^{(s)}$ , we can write the log-likelihood as

$$\mathcal{L}_\epsilon(\theta) \propto \log \sum_s \mathcal{N}(\mathbf{y}|f(\theta, \omega_s), \epsilon^2) \quad (16)$$

$$\approx -\log \epsilon - \frac{1}{2\epsilon^2} (\mathbf{y} - f(\theta, \omega_m))^2 \quad (17)$$

In the second line we have assumed  $\epsilon$  is very small and  $m$  is the index of the random seed producing the closest simulation to  $\mathbf{y}$ . For a finite difference approximation,

---

**Algorithm 1**  $\nabla U$  SPSA-ABC
 

---

**inputs:**  $\theta, d_\theta, f, \omega, \mathcal{L}_\epsilon, \pi, R$   
 $\hat{g} \leftarrow \mathbf{0}$   
**for**  $r = 1 : R$  **do**  
 $\Delta \sim 2 \cdot \text{Bernoulli}(1/2, D) - 1$   
**for**  $s = 1 : S$  **do**  
 $\mathbf{x}_+^{(s)} \leftarrow f(\theta + d_\theta \Delta, \omega_s)$   
 $\mathbf{x}_-^{(s)} \leftarrow f(\theta - d_\theta \Delta, \omega_s)$   
**end for**  
 $\hat{g} \leftarrow \hat{g} + \left( \mathcal{L}_\epsilon(\{\mathbf{x}_+^{(s)}\}) - \mathcal{L}_\epsilon(\{\mathbf{x}_-^{(s)}\}) \right) \cdot \Delta^{-1}$   
**end for**  
 $\hat{g} \leftarrow \hat{g} / (2d_\theta R) + \nabla \log \pi(\theta)$   
**return**  $-\hat{g}$

---

$\partial \mathcal{L}_\epsilon(\theta) / \partial \theta$  is

$$\frac{1}{4d_\theta \epsilon^2} \left( (\mathbf{y} - f(\theta - d_\theta, \omega_m^-))^2 - (\mathbf{y} - f(\theta + d_\theta, \omega_m^+))^2 \right) \quad (18)$$

On the other hand, the synthetic-likelihood is stable; using a deterministic representation, we have

$$\mu_\theta = \frac{1}{S} \sum_s f(\theta, \omega_s) \quad \sigma_\theta^s = \frac{1}{S-1} \sum_s (\mu_\theta - f(\theta, \omega_s))^2 \quad (19)$$

the gradients (for a 1-dim problem) use  $\epsilon$  as a smoothness prior in  $\partial \mathcal{L}_\epsilon(\theta) / \partial \theta$ :

$$-\frac{1}{2} \log \left( \frac{\sigma_{\theta^+}^2 + \epsilon^2}{\sigma_{\theta^-}^2 + \epsilon^2} \right) - \frac{(\mathbf{y} - \mu_{\theta^+})^2}{2(\sigma_{\theta^+}^2 + \epsilon^2)} + \frac{(\mathbf{y} - \mu_{\theta^-})^2}{2(\sigma_{\theta^-}^2 + \epsilon^2)} \quad (20)$$

In Figure 2, as part of our demonstration of HABC, we compare the gradient approximations around the true  $\theta_{\text{MAP}}$  using SL versus kernel- $\epsilon$  for a simple problem. Although there is a small bias using SL due to its Gaussian assumption, it has much smaller variance, convergence to this (biased) posterior should be stable. Further, [19] showed that convergence for SGHD type algorithms depends on the tails of the log-posterior, which suggests that despite its bias, the non-heavy tails of the Gaussian may allow SL to produce a more efficient Markov chain.

### 4.3 From Finite Differences to Simultaneous Perturbations

If the dimension of  $\theta$  is small, then *finite difference stochastic approximation* (FDSA) [9] can be applied to  $\nabla U(\theta)$  (conditioned on random seeds  $\omega$ ). The number of simulations required for FDSA is  $2SD$ , which may be acceptable for some small ABC problems. Our main goal is to scale ABC to high-dimensions and for that we need an alternative stochastic approximation to  $\nabla U(\theta)$ .

In the gradient-free setting, Spall [21, 22] provides a stochastic approximate to the true gradient using only 2

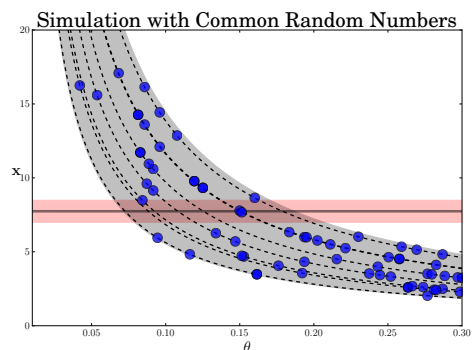


Figure 1: A view of a simulator using persistent random numbers; in other contexts, these are called common random numbers [10]. The horizontal line represents  $\mathbf{y}$  and red shading  $\pm 2\epsilon$ . The shaded curved region represents  $2\sigma$  of  $\pi(\mathbf{x}|\theta)$ . The dashed lines are  $f(\theta, \omega_s)$  for several values of  $\omega$ . The blue circles are potential random samples from  $\pi(\mathbf{x}|\theta)$ . For a fixed value  $\omega_s$ , the simulator produces deterministic outputs that change smoothly, even though the simulator itself is quite noisy.

forward simulations for any dimension  $D$  (though the approximation can be improved by averaging  $R$  estimates). Spall's *simultaneous perturbation stochastic approximation* (SPSA) algorithm works as follows. Let  $L$  be the gradient-free function we wish to optimize. Each approximation randomly generates a *perturbation mask* (our name)  $\Delta$  of dimension  $D$  where entry  $\Delta_d \sim 2\text{Bernoulli}(1/2) - 1$  (i.e. all entries randomly set to  $\pm 1$ ). Then  $L$  is evaluated at  $\theta + d_\theta \Delta$  and  $\theta - d_\theta \Delta$ , giving the gradient approximation  $\hat{g}(\theta) \approx \partial L(\theta) / \partial \theta$ :

$$\hat{g}(\theta) = \frac{L(\theta + d_\theta \Delta) - L(\theta - d_\theta \Delta)}{2d_\theta} \begin{bmatrix} 1/\Delta_1 \\ 1/\Delta_2 \\ \vdots \\ 1/\Delta_D \end{bmatrix} \quad (21)$$

If we let  $\hat{g}_r(\theta)$  be the estimate using perturbation mask  $\Delta_r$ , the estimate  $\hat{g}(\theta)$  can be improved by averaging  $\hat{g}(\theta) = 1/R \sum_r \hat{g}_r(\theta)$ . Algorithm 1 shows SPSA to estimate  $\nabla U(\theta)$ . The number of simulations required for SPSA is  $2SR$ , where  $R \geq 1$ .

Variations of SPSA include *one-sided* SPSA [22] (we use what Spall calls 2SPSA) and an algorithm for estimating the Hessian based on the same principle as SPSA [23]. The one-sided version is attractive computationally, but for HABC, the updates for  $\theta$  require simulating two-sides anyway (once at  $\theta$ , after a step is taken, and once for the one-sided gradient). SPSA has also been used within a procedure for maximum-likelihood estimation for hidden Markov models using ABC [8].

### 4.4 Persistent Random Numbers

The usefulness of applying *persistent* random numbers (PRNs) in SPSA has been previously demonstrated [10]. In

that work, the same random numbers are used to simulate both sides of the optimization function within the SPSA gradient. This makes sense intuitively, as we would generally assume that the expected simulation function varies smoothly in  $d\theta$ ; by using PRNs, this smoothness is easily exploited (see Figure 1). If we were to apply SPSA to Bayesian learning, then using PRNs in the gradient step would be analogous to using the same mini-batch for both sides of the computation. In the case where the number of random numbers is unknown or is itself random, we can simply consider seeds of the random number generator instead of vectors of random numbers.

In addition to using PRNs in simulations for each gradient computation, we have found that using PRNs helps HABC explore the parameter landscape more easily for some algorithms and problems. Intuitively, for a gradient-based sampling algorithm, it means a particle can slide along a smooth Hamiltonian landscape because the additive noise is suppressed. This is very similar to using dependent random streams to drive MCMC [15, 17], the main difference we believe is that we are using the Hamiltonian dynamics to drive proposals for  $\theta$  and using persistent seeds  $\omega$  to suppress simulation noise. The full benefits of suppressing the noise may be limited, however. Recent work has shown that scaling HMC for large data applications may be fundamentally limited [4]: noise from mini-batches causes biases in trajectories, which require either increasing mini-batch sizes (in our case, running more simulations) or decreasing the step size.

Using random seeds (versus, say, a set of random numbers) allows us to treat the simulator as a black-box, setting the random seed of its RNG without knowing the internal mechanisms it uses to generate random numbers. In light of our arguments above, we propose including persistent random seeds  $\omega$  in the state of our Markov chain. We will now describe a simple Metropolis-Hastings transition operator that randomly proposes *flipping* each seed  $\omega_s$  at time  $t$  with some probability  $\gamma$ .

This Metropolis-Hastings transition conditions of the current parameter location  $\theta$  and proposes changing a single random seed  $\omega$  (it easily generalizes to  $S$  seeds). The procedure is as follows: 1) propose a new seed  $\omega' \sim q(\omega'|\omega) = \pi(\omega)$  (independent of the current seed and from its uniform prior); 2) simulate *deterministically*  $\mathbf{x}' = f(\theta, \omega')$ ; 3) compute the acceptance ratio (which reduces to the ratio of  $\pi(\mathbf{y}|\mathbf{x}')/\pi(\mathbf{y}|\mathbf{x})$ ). It is straightforward to show that this leaves the target distribution invariant. The probability of the proposal is  $q(\omega', \omega|\theta, \omega) = \pi(\omega')\delta(\mathbf{x}' - f(\theta, \omega'))$ , where  $\delta(a)$  is a delta function at  $a = 0$ . Because the  $q$  has this form, the acceptance ratio simplifies:

$$\frac{\pi_\epsilon(\mathbf{y}|\mathbf{x}')\pi(\omega')\pi(\mathbf{x}'|\theta, \omega')\pi(\omega)\delta(\mathbf{x} - f(\theta, \omega))}{\pi_\epsilon(\mathbf{y}|\mathbf{x})\pi(\omega)\pi(\mathbf{x}|\theta, \omega)\pi(\omega')\delta(\mathbf{x}' - f(\theta, \omega'))} = \frac{\pi_\epsilon(\mathbf{y}|\mathbf{x}')}{\pi_\epsilon(\mathbf{y}|\mathbf{x})} \quad (22)$$

In pseudo-marginal ABC-MCMC one could propose  $q(\mathbf{x}^{(s)}|\theta)$  (fixing  $\theta$ ) and still sample correctly from the distribution of simulations with high likelihood at  $\theta$ . What we propose is slightly different. By instead keeping the random seeds fixed, we can sample  $\theta$  using HABC and use  $\omega$  as PRNs within the gradient computation step and suppress gradient noise over time. In this way, random seeds carry over the same additive noise from one step to the next.

## 5 Demonstration

We use a simple  $D = 1$  problem to demonstrate HABC. Let  $y = \frac{1}{N} \sum_i e_i$ , where  $e_i \sim \text{Exp}(1/\theta^*)$ ;  $\theta^* = 0.15$ ,  $N = 20$ , and  $y = 7.74$  in our concrete example. Assuming  $\pi(\theta) = \text{Gamma}(\alpha, \beta)$ , the true posterior is a gamma distribution with shape  $\alpha + N$  and rate  $\beta + Ny$ . Our simulator therefore generates the average of  $N$  exponential random variates with rate  $\lambda = 1/\theta$ . Data  $x \stackrel{\text{sim}}{\sim} \pi(x|\theta)$  are shown in Figure 1. We have explicitly shown the smoothness of the simulator by generating data along trajectories of fixed seeds  $\omega_s$ ; i.e. for several  $\omega_s$  we vary  $\theta$  (dashed lines are function  $f(\theta, \omega_s)$ ) and randomly reveal simulation data (blue circles). The horizontal line with shading indicates  $y \pm 2\epsilon$ , where  $\epsilon = 0.37$  is used throughout the demonstration.

### 5.1 Bias and Variance of $\nabla \hat{U}(\theta)$

To test our assumption that the synthetic-likelihood model is better suited for HABC, we ran FDSA at the true  $\theta_{\text{MAP}}$ . Using  $S = 5$  and  $S = 50$  and fixing  $\epsilon = 0.37$ , we gather  $10K$  gradient samples using kernel- $\epsilon$  and SL likelihoods. These gradient estimate densities are shown in Figure 2. An unbiased estimate of the gradient should be centered at 0. There are two important results. First, the SL estimates have a small bias, even at  $S = 50$ . This is because it is estimating the true Gamma distribution of  $\pi(\mathbf{x}|\theta)$  with a Gaussian. We can analytically estimate this bias as  $S \rightarrow \infty$ ; for this example it is  $-7.8$  which is what SL estimates are centered around ( $-9.3$  for  $S = 5$  and  $7.3$  for  $S = 50$ ). The kernel- $\epsilon$  likelihood, on the other hand, exhibits low bias at  $S = 50$ . However, the second important result is the variances. SL variances decrease quickly with  $S$ :  $\sigma^2 = 43^2 \rightarrow 4.9^2$ , whereas kernel- $\epsilon$  starts very high and remains high:  $\sigma^2 = 147^2 \rightarrow 19^2$ . It is for this reason that we have chosen to use SL likelihoods for our gradient estimates, despite their small bias. As mentioned in Section 4.2 it is possible that other likelihood models, such as a kernel density estimate, might provide low bias and low variance gradient estimates. We leave this for future work.

### 5.2 Posterior Inference using HABC

We ran chains of length  $50K$  for SL-MCMC, SGLD, SGHMC, and SGNHT versions of HABC using SL gra-

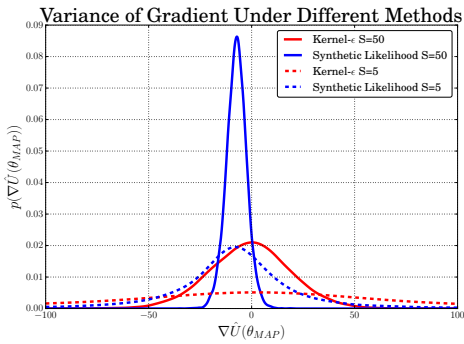


Figure 2: Variance of gradient estimation using kernel- $\epsilon$  and SL for different values of  $S \in \{5, 50\}$  and fixed  $\epsilon = 0.37$  (the same used in the other results). When  $S = 5$ , the empirical estimates of  $\nabla \hat{U}(\theta_{MAP})$  are  $-12 \pm 147$  (kernel- $\epsilon$ ) and  $-9.3 \pm 43$  (SL). When  $S = 50$  they are  $-0.80 \pm 19$  (kernel- $\epsilon$ ) and  $-7.3 \pm 4.9$  (SL). Note the large discrepancy in variance. Note the limit of  $S \rightarrow \infty$ ,  $\nabla \hat{U}(\theta_{MAP}) = -7.8$ . The bias if SL gradients is due to its Gaussian approximation (smoothed by  $\epsilon$ ) of  $\pi(\mathbf{x}|\theta)$ , which is a heavy-tailed Gamma distribution (the sum of  $N$  exponentials).

dient estimates ( $S = 5$ ). SL-MCMC refers to pseudo-marginal ABC-MCMC. We note that SGHMC gave results nearly identical to SGNHT, so are not shown due to space limitations. In one set of experiments, the same random seeds were used for gradient computations but did not persist over time steps; these experiments are called *non-persistent*. In another set of runs, we resampled  $\omega_s$  at each time step with probability  $\gamma = 0.1$ ; these experiments are *persistent*. In Figure 3 we show the posterior distributions for these experiments; in Table 1 we report the *total variational distance* between the true posterior and the ABC posteriors using the first  $10K$  samples and after  $50K$  samples (averaged over 5 chains). Of note is the poor approximation of SG-Thermostats when the seeds are not persistent. By adding persistent seeds, SGNHT gives similar posteriors to the other methods.

In Figure 4 we show the trace plots of the last 1000 samples from a single chain for each algorithm. In the left column, traces for non-persistent random seeds are shown, and on the right, traces for persistent seeds. We can observe that persistent random seeds further reduces the random walk behavior of all three methods. We also observe small improvements in total variational distance for SL-MCMC and SGLD, while SGNHT improves significantly. We find this a compelling mystery. Is it because of the interaction between hyperparameters and stochastic gradients, or is this an artifact of this simple model?

## 6 Experiments

We present experimental results comparing HABC with standard ABC-MCMC for two challenging simulators. The first is the *blowfly* model which uses stochastic differential equations to model possibly chaotic population dynamics

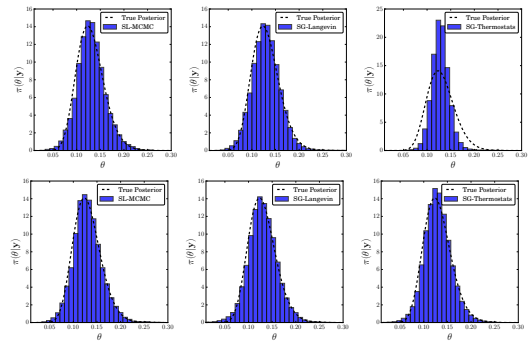


Figure 3: Posterior distributions for the demonstration problem; columns left to right: SL-MCMC, SGLD (SG-Langevin), SGNHT (SG-Thermostats). **Top row:** No persistent seeds. **Bottom row:** Persistent seeds with  $\gamma = 0.1$ . Histograms of the posterior estimates are overlaid with the true posterior (dashed line). All algorithms (except for SGNHT for non-persistent  $\omega$ ) give roughly the same posterior estimate. By adding persistent  $\omega$  SGNHT achieved similar posteriors to the other algorithms.

Table 1: Average total variational distance (tvd) for the demonstration problem. *Non-persistent* used no persistent random seeds, whereas *Persistent* randomly proposes a new  $\omega_s$  with  $\gamma = 0.1$ . Each algorithms' parameters were optimized for minimal tvd after  $10K$  samples. The results for SGHMC (not shown) and SGNHT are nearly identical.

| Algo    | Non-persistent |       | Persistent |       |
|---------|----------------|-------|------------|-------|
|         | 10K            | 50K   | 10K        | 50K   |
| SL-MCMC | 0.047          | 0.045 | 0.045      | 0.045 |
| SGLD    | 0.049          | 0.048 | 0.048      | 0.043 |
| SGNHT   | 0.232          | 0.239 | 0.055      | 0.051 |

[28]. Although it is a low-dimensional problem, the noise and chaotic behavior of the model make it challenging for gradient-based sampling. Our second experiment applies HABC to a Bayesian logistic regression model. Although we only use 2 classes (0's versus 1's), the dimensionality is very high ( $D = 1568$ ). We show that HABC can work well despite using SPSA gradients.

### 6.1 Blowfly

For these experiments, a simulator of adult sheep blowfly populations [28] is used with statistics set to those from [14]. The observational vector  $\mathbf{y}$  is a time-series of a fly population counted daily. The population dynamics are modeled using a stochastic differential equation<sup>1</sup>

$$N_{t+1} = PN_{t-\tau} \exp(-N_{t-\tau}/N_0)e_t + N_t \exp(-\delta\epsilon_t)$$

where  $e_t \sim \mathcal{G}(1/\sigma_p^2, 1/\sigma_p^2)$  and  $\epsilon_t \sim \mathcal{G}(1/\sigma_d^2, 1/\sigma_d^2)$  are sources of noise, and  $\tau$  is an integer. In total, there are  $D =$

<sup>1</sup>Equation 1 in Section 1.2.3 of the supplementary information in [28].

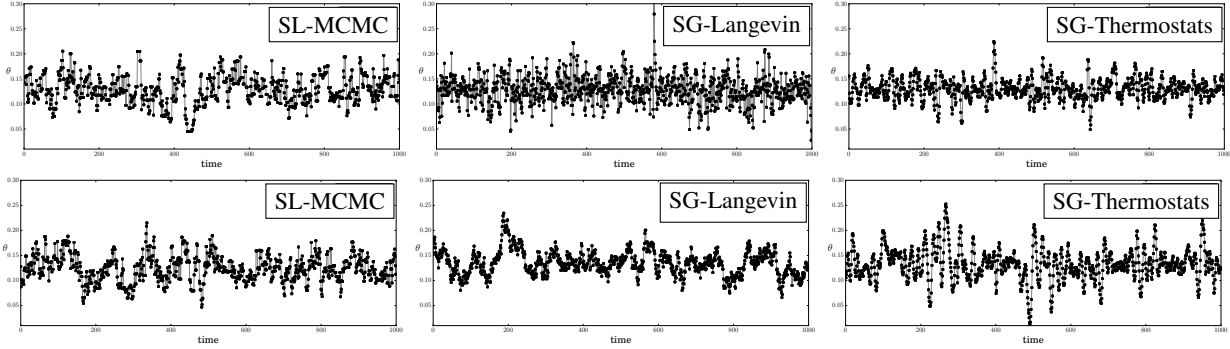


Figure 4: Trajectories of the last 1000  $\theta$  samples for the demonstration problem. **Top row:** Non-persistent random seeds. **Bottom row:** Persistent random seeds with  $\gamma = 0.1$ . Each algorithm’s parameters were optimized to minimize the total variational distance. With persistent seeds, each algorithm’s random walk behavior is suppressed. Without persistent seeds, the optimal step-size  $\eta$  for SGNHT is small, resulting in an under-dispersed estimate of the posterior; when the seeds are persistent, the gradients are more consistent, and the optimal step-size is larger and therefore there is larger injected noise. The resulting posteriors are shown in Figure 3.

6 parameters  $\theta = \{\log P, \log \delta, \log N_0, \log \sigma_d, \log \sigma_p, \tau\}$ . As [14] we place broad log-normal priors over  $\theta_{1..5}$  and a Poisson prior over  $\tau$ . This is considered a challenging problem because slight changes to some parameter settings can produce degenerate  $\mathbf{x}$ , while others settings can be very noisy due to the chaotic nature of the equations. The statistics from [14] are used ( $J = 10$ ): the log average of 4 quantiles of  $N/1000$ , the average of 4 quantiles of the first-order differences in  $N/1000$ , and the number of maximal population peaks under two different thresholds.

We compare difference HABC algorithms with ABC-MCMC for the blowfly population problem. We use  $\epsilon = \{1/2, 1/2, 1/2, 1/2, 1/4, 1/4, 1/4, 1/4, 3/4, 3/4\}$  (slightly different  $\epsilon$  from [14]) and  $S = 10$  for all experiments (this means that there are  $S$  random seeds). We use SPSA with  $R = 2$  using SL log-likelihoods for all HABC gradient estimates. Without persistent seeds, the number of simulations per time-step is  $2SR$  (about double marginal ABC-MCMC) and with it is  $2SR + 2S\gamma$ .

Figure 5 show the posterior distributions for three parameters for SL-MCMC, SGLD, and SGNHT using non-persistent seeds (persistent seeds, not shown, produced very similar posteriors). In the second row we show the trajectories of two parameters, clearly showing the suppressed random walk behavior of SGLD and SG-Thermostats relative to ABC-MCMC. In Figure 6 the scatter plots of trajectories are shown for two parameters. Though not shown due to space limitations, we have found that persistent seeds can improve convergence of the posterior predictive distribution. Further experiments with persistent seeds needs to be carried out to understand the extent to which the help and how to determine when they are necessary, if at all.

## 6.2 Bayesian Logistic Regression

We perform Bayesian inference on a logistic regression model using the digits 0 and 1 from MNIST. Although

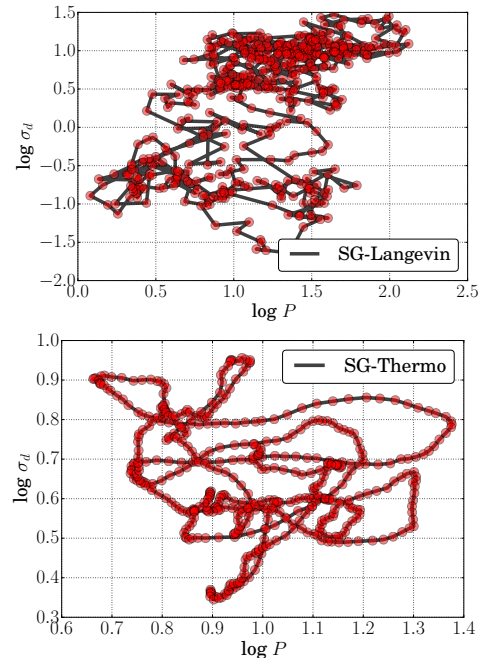


Figure 6: Blowfly trajectories of two parameters over the last 1000 time-steps. **Top:** SGLD and **Bottom:** SGNHT (SG-Thermostats). Relative to SL-MCMC (not shown), the Hamiltonian dynamics clearly show persistent  $\theta$  trajectories.

not technically an ABC problem because we use the actual likelihoods, it still represents a high-dimensional problem ( $D = 1568$ ) and is therefore useful to evaluate the potential of SPSA-like gradients in actual HABC problems. We first ran stochastic gradient descent to determine  $\theta_{\text{MAP}}$  using the true gradient. We then ran SGLD and SGNHT starting  $\theta_{\text{MAP}}$  to discover how well the algorithms explore the posterior. We examine how SGLD and SGNHT trajectories are affected by using SPSA instead of the *true* gradients. We use  $n = 100$  sized mini-batches and  $R = 10$  perturbations for SPSA. Figure 7 shows samples randomly projected onto 2 dimensions (1000 evenly sub-sampled from

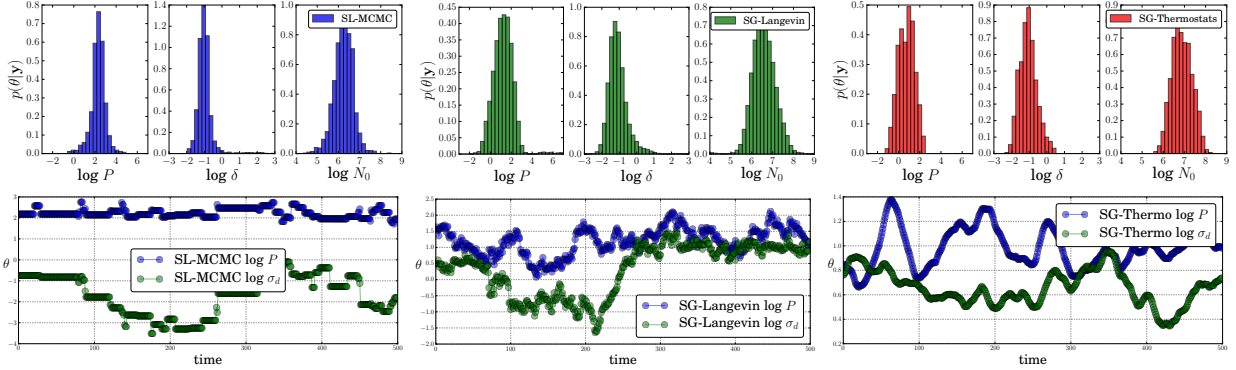


Figure 5: Blowfly posterior distributions (non-persistent seeds). **Top row:** Posteriors for three parameters for SL-MCMC (left set of three), SGLD (SG-Langevin) (middle), and SGNHT (SG-Thermostats) (right). **Bottom row:** Last trajectories of the last 1000 samples for two parameters for the same algorithms.

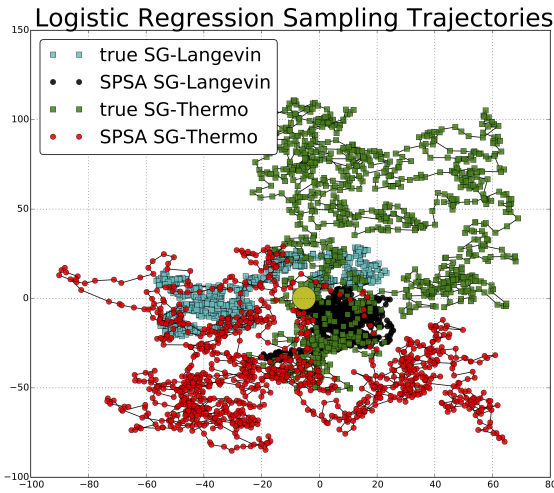


Figure 7: Bayesian logistic regression sampling trajectories randomly projected. The yellow circle is the projected MAP of  $\theta$ .

10K). We can clearly see that the trajectories using SPSA exhibit very similar behavior to Bayesian learning with the true gradients. This is very positive result that indicates HABC can successfully exploit the noisy and less informative gradients of SPSA.

## 7 DISCUSSION AND CONCLUSION

Hamiltonian ABC proposes a new set of algorithms for Bayesian inference of likelihood-free models. HABC builds upon the connections between Hamiltonian Monte Carlo with stochastic gradients and well-established gradient approximations based on a minimal number of forward simulations, even in high-dimensions. We have performed some preliminary experiments showing the feasibility of running HABC on both small and large problems, and we hope that the door has been opened for exploration of larger simulation-based models using HABC.

Another innovation we introduce is the use of persistent random seeds to suppress the simulator noise and therefore smooth the simulation landscape over a local region

of parameter space. For some algorithms run on certain models, improved performance has been observed. This is most likely to be the case for simulators with large additive noise and algorithms that benefit from long Hamiltonian trajectories (i.e. SGHMC and SGNHT). We feel that new classes of ABC algorithms could develop from using persistent random seeds, not just gradient-based samplers but traditional ABC-MCMC.

There are several unresolved and open questions regarding the application of stochastic gradients to ABC. The first issue is the importance of the bias-variance relationship for different ABC likelihood models. We found that using gradients based on the synthetic-likelihood greatly reduced their variance, but introduced a small bias, because of its Gaussian assumption. The second issue is setting algorithm parameters, in particular the step-sizes  $\eta$ , the injected noise  $C$  (for SGHMC/SGNHT), and the number of SPSA repetitions  $R$ . All of these parameters are highly interactive. Can we use statistical tests during the MCMC run to determine  $R$ ? Should  $\eta$  and  $C$  be set differently in the ABC setting? One final issue is monitoring or determining whether the correct amount of noise is being injected to ensure proper sampling. In SGLD [25], for example, we can always turn down  $\eta$  so that the injected noise term dominates, but when our goal is efficient exploration of the posterior, this is not a very satisfying solution.

Expensive simulators are an important class of models that we do not address in this work. However, previous work in Bayesian inference has shown the usefulness of HMC-based proposals based on Gaussian process of log-likelihood surfaces [18]. We could similarly use HABC with ABC surrogate models [14, 27] to minimize simulation calls, yet still benefit from Hamiltonian dynamics.

## Acknowledgements

We thank the anonymous reviewers for the many useful comments that improved this manuscript. MW acknowledges support from Facebook, Google, and Yahoo.



## References

- [1] Alquier, Pierre, Friel, Nial, Everitt, Richard, and Boland, Aidan. Noisy Monte Carlo: Convergence of Markov chains with approximate transition kernels. *Statistics and Computing*, pp. 1–19, 2014.
- [2] Andrieu, C. and Roberts, G. The pseudo-marginal approach for efficient Monte Carlo computations. *The Annals of Statistics*, 37(2):697–725, 2009.
- [3] Beaumont, Mark A, Zhang, Wenyang, and Balding, David J. Approximate Bayesian computation in population genetics. *Genetics*, 162(4):2025–2035, 2002.
- [4] Betancourt, MJ. The Fundamental Incompatibility of Hamiltonian Monte Carlo and Data Subsampling. *Journal of Machine Learning Research*, 37, 2015.
- [5] Chen, Tianqi, Fox, Emily B, and Guestrin, Carlos. Stochastic gradient Hamiltonian Monte Carlo. 2014.
- [6] Ding, Nan, Fang, Youhan, Babbush, Ryan, Chen, Changyou, Skeel, Robert D, and Neven, Hartmut. Bayesian sampling using stochastic gradient thermostats. In *Advances in Neural Information Processing Systems*, pp. 3203–3211, 2014.
- [7] Duane, Simon, Kennedy, Anthony D, Pendleton, Brian J, and Roweth, Duncan. Hybrid Monte Carlo. *Physics letters B*, 195(2):216–222, 1987.
- [8] Ehrlich, Elena, Jasra, Ajay, and Kantas, Nikolas. Gradient Free Parameter Estimation for Hidden Markov Models with Intractable Likelihoods. *Methodology and Computing in Applied Probability*, pp. 1–35, 2013.
- [9] Kiefer, Jack, Wolfowitz, Jacob, et al. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, 23(3):462–466, 1952.
- [10] Kleinman, Nathan L, Spall, James C, and Naiman, Daniel Q. Simulation-based optimization with stochastic approximation using common random numbers. *Management Science*, 45(11):1570–1578, 1999.
- [11] Leimkuhler, Benedict and Reich, Sebastian. A metropolis adjusted Nosé-Hoover thermostat. *ESAIM: Mathematical Modelling and Numerical Analysis*, 43(04):743–755, 2009.
- [12] Marin, J.-M., Pudlo, P., Robert, C.P., and Ryder, R.J. Approximate bayesian computational methods. *Statistics and Computing*, 22:1167–1180, 2012.
- [13] Marjoram, Paul, Molitor, John, Plagnol, Vincent, and Tavaré, Simon. Markov chain Monte Carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 100(26):15324–15328, 2003.
- [14] Meeds, Edward and Welling, Max. GPS-ABC: Gaussian process surrogate approximate bayesian computation. *Uncertainty in AI*, 2014.
- [15] Murray, Iain and Elliott, Lloyd T. Driving Markov chain Monte Carlo with a dependent random stream. *arXiv:1204.3187*, 2012.
- [16] Neal, Radford M. MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2, 2011.
- [17] Neal, Radford M. How to View an MCMC Simulation as a Permutation, with Applications to Parallel Simulation and Improved Importance Sampling. *Technical Report No. 1201, Dept. of Statistics, University of Toronto*, 2012.
- [18] Rasmussen, C.E. Gaussian processes to speed up hybrid monte carlo for expensive bayesian integrals. *Bayesian Statistics*, 7:651–659, 2003.
- [19] Roberts, Gareth O and Tweedie, Richard L. Exponential convergence of Langevin distributions and their discrete approximations. *Bernoulli*, pp. 341–363, 1996.
- [20] Sisson, Scott A and Fan, Yanan. Likelihood-free markov chain monte carlo. *Arxiv preprint arXiv:1001.2058*, 2010.
- [21] Spall, James C. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *Automatic Control, IEEE Transactions on*, 37(3):332–341, 1992.
- [22] Spall, James C. Adaptive stochastic approximation by the simultaneous perturbation method. *Automatic Control, IEEE Transactions on*, 45(10):1839–1853, 2000.
- [23] Spall, James C. Monte Carlo computation of the Fisher information matrix in nonstandard settings. *Journal of Computational and Graphical Statistics*, 14(4), 2005.
- [24] Turner, Brandon M. and Sederberg, Per B. A generalized, likelihood-free method for posterior estimation. *Psychonomic Bulletin & Review*, 21(2):227–250, 2014.
- [25] Welling, Max and Teh, Yee W. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pp. 681–688, 2011.
- [26] Wilkinson, R. Approximate Bayesian computation (ABC) gives exact results under the assumption of model error. *Statistical Applications in Genetics and Molecular Biology*, 12(2):129–142, 2013.
- [27] Wilkinson, R. Accelerating abc methods using gaussian processes. *AISTATS*, 2014.
- [28] Wood, Simon N. Statistical inference for noisy nonlinear ecological dynamic systems. *Nature*, 466(7310):1102–1104, 2010.

---

# (Nearly) Optimal Differentially Private Stochastic Multi-Arm Bandits

---

**Nikita Mishra**

Department of Computer Science  
University of Chicago  
nmishra@cs.uchicago.edu

**Abhradeep Thakurta**

Yahoo! Research, Sunnyvale  
guhathakurta.abhradeep@gmail.com

## Abstract

We study the problem of private stochastic multi-arm bandits. Our notion of privacy is the same as some of the earlier works in the general area of private online learning [13, 17, 24]. We design algorithms that are i) differentially private, and ii) have regret guarantees that (almost) match the regret guarantees for the best non-private algorithms (e.g., upper confidence bound sampling and Thompson sampling). Moreover, through our experiments, we empirically show the effectiveness of our algorithms.

## 1 INTRODUCTION

A general abstraction of bandit problems is the following: Given a set of  $k$  arms  $\mathcal{C} = \{a_1, \dots, a_k\}$ , at each time step one pulls an arm  $a_i \in \mathcal{C}$ , gets a reward corresponding to  $a_i$  and the objective of the algorithm is to obtain large cumulative reward over all time steps  $T$ . In a class of problems called the *adversarial bandits*, it is assumed that the reward can be adversarial, in the sense that the rewards for each of the arms in  $\mathcal{C}$  are arbitrarily chosen. The other class of problems is called the *stochastic bandits* where it is assumed that the reward for each of the arm in  $\mathcal{C}$  is from an unknown distribution. (See [5] for a detailed introduction to these classes.) In this paper we are interested in the stochastic bandit setting. We analyze two of the most common algorithms in this setting in the context of differential privacy, *upper confidence bound* (UCB) sampling by [4] and *Thompson sampling* by [2, 18]. We show that one can modify UCB sampling and Thompson sampling algorithms in such a way that ensures: i) differential privacy, and ii) regret guarantee which is only poly  $\log T$  factor worse compared to the regret for the non-private variants.

We now focus on the semantics of differential privacy in this setting where the data points (the rewards) arrive online in a stream at every time step. This setting was first studied by [13] and then followed by [17] and [24]. Let

$f_t = \langle f_t(a_1), \dots, f_t(a_k) \rangle$  be the vector of rewards for all the arms in  $\mathcal{C}$  at time step  $t$ . Privacy guarantee will ensure that from the output of the algorithm over all the  $T$  time steps the adversary will not be able to distinguish between the presence or absence of any single reward vector  $f_t$ . [16] studied differentially private online algorithms in the *full-information* setting, where at each time step  $t$  the algorithm can see the complete reward vector  $f_t$  as opposed to  $f_t(a)$  for the arm  $a$  pulled in the bandit setting. [24] extended this line of work to obtain tighter and nearly optimal regret guarantees for both full-information and adversarial bandit settings. Recall that in the non-private world, the full-information and the adversarial bandit settings both have optimal regret guarantee of  $\Omega(\sqrt{T})$  (see [23]). In contrast, stochastic bandit algorithms enjoy a regret of  $O(\log T)$ . In this work we obtain the *first* and *nearly optimal* regret guarantees for stochastic bandit problems. Since, stochastic bandit algorithms have a very different flavor than adversarial online algorithms, we needed to introduce new proof techniques tailored to our problem.

The stochastic multi-armed bandit algorithms usually run in the two implicit phases *exploration phase* and *exploitation phase*. During the exploration phase, the algorithm uses the pull of the arms in the initial rounds to get a sufficiently accurate estimate of the means of the arms, and then in the second phase uses this information to guide the decision of pulling of arms in the later rounds. However, in order to ensure differential privacy, we are required to introduce certain randomness in the observed rewards, but this tends to grossly corrupt the estimation of the means of the arms. At a high-level, for both UCB and Thompson sampling, we address this issue by increasing the number of rounds used by the algorithm to estimate these means. The exact details are very different for both the algorithms are discussed in the respective sections. One important point to keep in mind is that although we make stochastic assumptions on the data to ensure strong utility guarantees, we *do not* make any assumptions on the data while ensuring privacy for our algorithms. Using the distributional assumption on the data for any kind of privacy guarantee may be disastrous, since real world data may not fol-

low the assumed distribution. For our algorithms, privacy should hold in the worst case scenario but the utility guarantee holds under distributional assumptions on the data. Finally, to fortify our theoretical guarantees we show that they work well in experiments and often are competitive w.r.t. the non-private algorithms.

**Practical motivation.** In the past few years bandit algorithms have become extremely popular in both the theoretical and applied online learning community. Along with having strong theoretical guarantees, these algorithms have found wide applicability in Internet scale systems. A concrete usage scenario is in the online search advertisement industry. Companies like Google, Microsoft and Yahoo run multi-million dollar industry based on showing relevant advertisements for a web-queries. For a given search query by the user, the search engine displays a few advertisement which the user is most likely to click (commonly known as the *main-line advertisement*). After the advertisement gets displayed, the user chooses to either click or not click. If he/she clicks, then we say that the search engine gets a reward of one, and zero otherwise. To earn maximum revenue, the search engine strives to get high overall reward. One important feature of this setup (and generally in bandit learning) is that the search engine only gets to see the reward for the advertisements that were shown to the user, and gets no feedback about what the user would have done on other candidate advertisements. Bandit algorithms have been extremely successful in such settings and often enjoy strong theoretical guarantees for the overall reward. Settings like search advertisements immediately raise privacy concern for the users whose data get used in training the learning algorithms. Consider the example of a user who searches for a lawyer in Port Jefferson, NY and clicks on an advertisement of a divorce law firm. If the learning algorithm uses this feedback to show the same advertisement for similar query (e.g., finance lawyer in Port Jefferson, NY), then one can use such feedbacks to infer significant amount of information about a particular user. [19] on the Facebook advertisement recommendation system and [6] on the Amazon recommendation system, showed that one can leverage such side information to breach user privacy.

### 1.1 Our Contributions

Here, we provide an overview of our contributions.

- **Differentially private UCB sampling [Section 3].** We provide a differentially private variant of UCB sampling which enjoys the same utility guarantee as the non-private algorithm up to poly-logarithmic factors in the number of time steps  $T$ . The privacy guarantee follows via standard reduction to the *tree-based-aggregation scheme*, proposed by [14, 7]. Our utility analysis goes via carefully analyzing the *exploration phase* of the algorithm, where it estimates the means of the arms. Thus, we provide a version of UCB sampling algorithm which

is robust to noise.

- **Differentially private Thompson sampling [Section 4].** We also provide a differentially private variant of Thompson sampling which enjoys the same utility guarantee as the non-private algorithm up to poly-logarithmic factors in the number of time steps  $T$ . The privacy analysis is very similar to the UCB sampling. But, the utility analysis is much trickier. In fact even without privacy analyzing the *exploration phase* was considered extremely difficult and it took more than seventy years for the community to come up with a formal analysis (see [2, 18]). The main technical contribution of this section is to come up with a noise robust version of Thompson sampling. In order to obtain this robustness, the exploration phase of the original algorithm had to be modified significantly. A question that we leave open in this work is that if that is necessary.

## 2 BACKGROUND AND PROBLEM DEFINITION

### 2.1 Background on Differential privacy

In this section we provide a short overview of differential privacy.  $\mathcal{D} = \langle f_1, \dots, f_T \rangle$  be a data set of all the reward functions. We call a data set  $\mathcal{D}'$  neighbor of  $\mathcal{D}$  if it differs from  $\mathcal{D}$  in exactly one reward function. Let  $\mathcal{C}^T$  be the space of all  $T$  outputs of Algorithm  $\mathcal{A}$ .

**Definition 1** (Differential privacy [12]). *A randomized algorithm  $\mathcal{A}$  is  $\epsilon$ -differentially private if for any two neighboring data sets  $\mathcal{D}$  and  $\mathcal{D}'$ , and for all sets  $\mathcal{O} \subseteq \mathcal{C}^T$  the following holds:*

$$\Pr[\mathcal{A}(\mathcal{D}) \in \mathcal{O}] \leq e^\epsilon \Pr[\mathcal{A}(\mathcal{D}') \in \mathcal{O}].$$

As per the semantics of the definition, differential privacy ensures that an adversary gets to know “almost the same thing” about a reward function  $f_t$  irrespective of its presence or absence in the data set  $\mathcal{D}$ . This closeness is measured by the privacy parameter  $\epsilon$ . A typical choice of  $\epsilon$  is a small constant (e.g., 0.1). One important requirement of the definition is that the guarantee should hold for every pair of neighboring data sets. This directly implies that although for the regret analysis of our algorithm  $\mathcal{A}$  we can assume that the reward function comes from some underlying distribution, but we *cannot* use any stochastic assumption on the reward functions for privacy guarantee. Next, we discuss some of the basic tools for designing differentially private algorithms.

**Laplace and Gamma mechanism.** Laplace [12] and Gamma mechanism [10] are sensitivity based methods to achieve differential privacy. The best way to introduce Laplace mechanism is via the following setting. Consider a domain of data entries  $\mathcal{U}$  and a function  $f : \mathcal{U}^* \rightarrow \mathbb{R}$ . For the domain of data sets  $\mathcal{U}^n$ , we define the sensitivity of the

function  $f$  as below.

$$s = \text{Sensitivity}(f) = \max_{\text{Neighbors } \mathcal{D}, \mathcal{D}' \in \mathcal{U}^*} |f(\mathcal{D}) - f(\mathcal{D}')|$$

Let  $\text{Lap}(\lambda)$  be the Laplace distribution with scaling parameter  $\lambda$ , *i.e.*, the density function of this distribution is given by  $\frac{1}{2\lambda} e^{-|x|/\lambda}$ . Laplace mechanism states that for a given data set  $\mathcal{D}$  and noise  $N \sim \text{Lap}(\frac{s}{\epsilon})$ ,  $f(\mathcal{D}) + N$  is  $\epsilon$ -differentially private. The proof of this claim directly follows from the density function for Laplace distribution and triangle inequality. [12]

Gamma mechanism is also very similar to Laplace mechanism. The only difference being that we are now working with a vector valued function  $f : \mathcal{U}^* \rightarrow \mathbb{R}^p$ . Analogous to Laplace mechanism, let us define the  $L_2$ -sensitivity of the function  $f$  as below.

$$s = \text{Sensitivity}(f) = \max_{\text{Neighbors } \mathcal{D}, \mathcal{D}' \in \mathcal{U}^*} \|f(\mathcal{D}) - f(\mathcal{D}')\|_2$$

Gamma mechanism states that if we sample the noise vector  $N \in \mathbb{R}^p$  from the noise distribution with kernel  $e^{-\epsilon \|N\|_2/s}$ , then  $f(\mathcal{D}) + N$  is  $\epsilon$ -differentially private. (See [10] for the proof.)

**Tree based aggregation.** Initially proposed by [14, 7], this aggregation scheme is extremely effective in releasing private continual statistics over a data stream. To define the scheme formally, consider a data set  $\mathcal{D} = \langle f_1, \dots, f_T \rangle$ , where each entry  $f_t \in [0, 1]$ , and these entries arrive in a stream, *i.e.*, at every time step  $t \in [T]$ , one entry  $f_t$  arrives.

At every time step  $t$ , the task is to output  $v_t = \sum_{\tau=1}^t f_t$  while ensuring that the complete output sequence  $\langle v_1, \dots, v_T \rangle$  is  $\epsilon$ -differentially private. One can design an algorithm (using a binary tree based aggregation), which assures an additive error of  $O\left(\frac{\log^{1.5} T}{\epsilon}\right)$  per query. Moreover, it is simple to extend this scheme to the case where  $f_t \in \mathbb{R}^p$  and  $\|f_t\|_2 \leq 1$  for all  $t \in [T]$ . Since, the noise used in this scheme is exponential in nature, a high-probability guarantee is also immediate. We defer the details of the scheme to Appendix A. Suppose at every time step  $t \in [T]$ , one entry from dataset  $\mathcal{D}$ ,  $f_t \in [0, 1]$  arrives and the task is to output  $v_t = \sum_{\tau=1}^t f_t$  while ensuring that the complete output sequence  $\langle v_1, \dots, v_T \rangle$  is  $\epsilon$ -differentially private. This algorithm uses a binary tree based aggregation scheme, which assures an additive error of  $O\left(\frac{\log^{1.5} T}{\epsilon}\right)$  per query. We defer the details of the scheme to Appendix A. Moreover, it can be extended to the case where  $f_t \in \mathbb{R}^p$  and  $\|f_t\|_2 \leq 1$  for all  $t \in [T]$ .

## 2.2 Background on Stochastic Multi-arm Bandits

A typical setup for an online learning problem is as follows: There is a sequence of *reward functions*  $f_1, \dots, f_T$

arriving in a stream (*i.e.*, one at every time step  $t \in [T]$ ), where each  $f_t$  maps from some fixed set  $\mathcal{C}$  to  $\mathbb{R}$ . At every time step  $t$ , an online learning algorithm  $\mathcal{A}$  is expected to produce an element  $x_t \in \mathcal{C}$  before  $f_t$  is revealed to it. Once  $f_t$  gets revealed to  $\mathcal{A}$ , the algorithm pays a reward of  $f_t(x_t)$ . The objective of  $\mathcal{A}$  is to be competitive with the best choice of  $x \in \mathcal{C}$  in the hindsight, *i.e.*, be competitive with  $\max_{x \in \mathcal{C}} \sum_{t=1}^T f_t(x)$ . A natural measure of the utility of  $\mathcal{A}$  is

$$\text{regret, defined: } \text{Regret}_{\mathcal{A}}(T) = \max_{x \in \mathcal{C}} \sum_{t=1}^T f_t(x) - \sum_{t=1}^T f_t(x_t).$$

(For a detailed discussion, see [23])

There are two popular settings under which these problems are studied, namely, i) *online learning under complete feedback or the full-information setting*, and ii) *online learning under partial feedback or the bandit setting*. In the first setting, it is assumed that at time step  $t$  after the algorithm  $\mathcal{A}$  has produced  $x_t$ , it gets to see the complete reward function  $f_t$ . In the second setting, the algorithm gets much lesser information from the environment and just sees the evaluation of  $f_t$  at  $x_t$ .

## 2.3 Problem Definition

Let us assume that for all  $t \in [T]$  we have  $f_t : \mathcal{C} \rightarrow [0, 1]$ , where  $\mathcal{C}$  is the set of  $k$ -arms. Additionally we assume that for each arm  $a \in \mathcal{C}$ , each  $f_t(a)$  is an independent sample from a distribution with mean  $\mu_a$ . The objective is to design differentially private algorithms, whose regret (defined in (1)) depends poly-logarithmically in the number of reward functions  $T$ .

$$\mathbb{E} [\text{Regret}_{\mathcal{A}}(T)] = T \max_{a \in \mathcal{C}} \mu_a - \mathbb{E} \left[ \sum_{t=1}^T f_t(a(t)) \right]. \quad (1)$$

Here,  $a(t) \in \mathcal{C}$  is the arm played in the  $t$ -th time step.

## 3 PRIVATE UCB SAMPLING

Upper Confidence Bound (UCB) sampling by [4] is a heuristic for stochastic multi-arm bandit (MAB) problems, which despite being very simple gives strong utility guarantees. The regret for UCB  $O^*(\log T)$  in fact matches the asymptotic lower given by [20] upto a problem dependent constant. This is in sharp contrast with the algorithms for adversarial multi-arm bandit problems where the regret depends polynomially on the time horizon  $T$  (see [1, 15]). Recently [24] provided differentially private algorithms for adversarial bandit problems, which are almost optimal in the parameter  $T$ . In this section, for the UCB algorithm for stochastic MAB, we provide a differentially private algorithm whose expected regret is only poly-logarithmically worse in  $T$ . Before we move on to the differentially private UCB algorithm, we provide a brief overview of the non-private version of the algorithm.

**Background on UCB sampling.** Recall that in the MAB problem there are  $k$ -arms denoted by the set  $\mathcal{C}$ , and at each time step  $t$  each arm  $a \in \mathcal{C}$  produces either 0 or 1 from some unknown but fixed distribution on  $[0, 1]$  with mean  $\mu_a$ . The objective is to minimize the regret defined in (1). For each arm  $a$ , the UCB algorithm records the number of times it got pulled  $n_a(t)$  and the average reward  $\frac{r_a(t)}{n_a(t)}$  aggregated so far upto time  $t$ . Upon initialization, the algorithm pulls each arm exactly once. Later, the algorithm picks the arm with the highest upper confidence bound, i.e.,  $\arg \max_{a \in \mathcal{C}} \frac{r_a(t)}{n_a(t)} + \sqrt{\frac{2 \log t}{n_a(t)}}$ .

**Theorem 2** (Regret for non-private UCB Sampling [4]). *Let  $\mu^* = \max_{a \in \mathcal{C}} \mu_a$ . For each arm  $a \in \mathcal{C}$ , let  $\Delta_a = \mu^* - \mu_a$ . The expected regret of UCB sampling algorithm is as follows:*

$$\mathbb{E}[\text{Regret}_{\text{UCB}}(T)] = O\left(\sum_{a \in \mathcal{C}: \mu_a < \mu^*} \frac{\log T}{\Delta_a} + \Delta_a\right).$$

The expectation is over the randomness of the data.

### 3.1 Private UCB Sampling: Algorithm and Analysis

In Algorithm 1 we modify the UCB sampling algorithm to obtain an  $\epsilon$ -differentially private variant. Notice that for each arm  $a \in \mathcal{C}$  the average reward  $r_a(t)$ , is the only term that depends directly on the data set whose privacy we want to protect. So, if we can ensure that this sequence,  $r_a(t)$ ,  $t \in [T]$  is  $\epsilon/k$ -differentially private for each arm  $a$ , then immediately we have  $\epsilon$ -differential privacy for the complete algorithm. We invoke the tree based aggregation algorithm from Section 2.1 to make these sequences private. Additionally, to counter the noise added to the empirical mean, we loosen the confidence interval for the biases of each arm.

#### 3.1.1 Privacy Analysis

**Theorem 3** (Privacy guarantee). *Algorithm 1 is  $\epsilon$ -differentially private.*

*Proof.* The algorithm only accesses the reward for its computation via the tree based aggregation scheme (see Section 2). Since, there are  $k$ -arms, we maintain  $k$  separate trees, each of which is guaranteed to be  $\epsilon_0 = \frac{\epsilon}{k}$  differentially private. Using the composition property of differential privacy, we immediately conclude that Algorithm 1 is  $\epsilon$ -differentially private.  $\square$

#### 3.1.2 Regret Analysis

The expected regret of the algorithm is given by  $\mathbb{E}[\sum_{a \in \mathcal{C}: \mu_a < \mu_{a^*}} \Delta_a n_a(T)]$ . Hence, if our algorithm limits the pulls of the bad arms, we are done. Our regret analysis proceeds as follows, first we bound the amount of noise that

---

### Algorithm 1 Differentially Private UCB Sampling

---

- Input:** Time horizon:  $T$ , arms:  $\mathcal{C} = \{a_1, \dots, a_k\}$ , privacy parameter:  $\epsilon$ , failure probability:  $\gamma$ .
- 1: Create an empty tree  $\text{Tree}_{a_i}$  with  $T$ -leaves for each arm  $a_i$ . Set  $\epsilon_0 \leftarrow \epsilon/k$ .
  - 2: **for**  $t \leftarrow 1$  to  $k$  **do**
  - 3: Pull arm  $a_t$  and observe reward  $f_t(a_t)$ .
  - 4: Insert  $f_t(a_t)$  into  $\text{Tree}_{a_t}$  via *tree based aggregation* (see Section 2.1) with privacy parameter  $\epsilon_0$ .
  - 5: **Number of pulls:**  $n_{a_t} = 1$ .
  - 6: **end for**
  - 7: Confidence relaxation:  $\Gamma \leftarrow \frac{k \log^2 T \log((kT \log T)/\gamma)}{\epsilon}$ .
  - 8: **for**  $t \leftarrow k + 1$  to  $T$  **do**
  - 9: **Total reward:**  $r_a(t) \leftarrow$  Total reward computed using  $\text{Tree}_a$ , for all  $a \in \mathcal{C}$ .
  - 10: Pull arm  $a^* = \arg \max_{a \in \mathcal{C}} \left(\frac{r_a(t)}{n_a} + \sqrt{\frac{2 \log t}{n_a}} + \frac{\Gamma}{n_a}\right)$  and observe  $f_t(a^*)$ .
  - 11: **Number of pulls:**  $n_{a^*} \leftarrow n_{a^*} + 1$ .
  - 12: Insert  $f_t(a^*)$  into  $\text{Tree}_{a^*}$  using *tree based aggregation* and privacy parameter  $\epsilon_0$ .
  - 13: **end for**
- 

can be present in any of the total rewards  $r_a(t)$ . And later using this bound, we show that the number of times the suboptimal arms get pulled is small. We bifurcate the analysis of the each of the suboptimal arms into exploration and exploitation phase. We argue that in case of bad arms, after getting pulled for  $O\left(\frac{k \log^2 T \log(kT)}{\epsilon \Delta_a^2}\right)$  rounds the arm is not selected again with high probability. The main arguments in this analysis follow the general sequence of arguments in the analysis for non-private UCB sampling. (See [9] for a comparison.)

**Theorem 4** (Utility guarantee). *Let  $\{\mu_a : a \in \mathcal{C}\}$  be the biases of the  $k$ -arms in the set  $\mathcal{C}$ . Let  $\mu^* = \max_{a \in \mathcal{C}} \mu_a$  and for each arm  $a \in \mathcal{C}$ ,  $\Delta_a = \mu^* - \mu_a$ . With probability at least  $1 - \gamma$  (over the randomness of the algorithm), the expected regret (over the randomness of the data) is as follows:*

$$\mathbb{E}[\text{Regret}_{\text{Priv-UCB}}(T)] = O\left(\sum_{a \in \mathcal{C}: \mu_a < \mu^*} \frac{k \log^2 T \log(kT/\gamma)}{\epsilon \Delta_a} + \Delta_a\right).$$

In the following lemma, we bound the error in the computation of the total reward in Line 9 in Algorithm 1, introduced due to privacy.

**Lemma 5.** *For all arms  $a \in \mathcal{C}$  and all time step  $t \in [T]$ , w.p.  $\geq 1 - \gamma$  (over the randomness of the algorithm), the error in the computation of the total reward for a till time  $t$  is at most  $\frac{k \log^2 T \log((kT \log T)/\gamma)}{\epsilon}$ .*

The proof of this lemma is given in Appendix B.

**Lemma 6.** For a given arm  $a \in \mathcal{C}$ , if the mean  $\mu_a < \mu^*$  and  $\Delta_a = \mu^* - \mu_a$ , then w.p.  $\geq 1 - \gamma$  (over the randomness of the algorithm)  $\mathbb{E}[n_a(T)] = O\left(\frac{\log^2 T \log(kT/\gamma)}{\epsilon \Delta_a^2}\right) + \zeta$ . Here  $\zeta$  is a fixed positive constant independent of the problem parameters. The expectation is over the randomness of the data and  $O(\cdot)$  only hides multiplicative constants.

*Proof.* This lemma provides us with an upper limit on the expected number of pulls of the suboptimal arms. For each arm we partition the analysis into two phases (exploration phase and exploitation phase). For an arm  $a$ , such that  $\mu_a < \mu^*$ , we show that the exploration phase lasts for  $s_a = \frac{8k \log^2 T \log((kT \log T)/\gamma)}{\epsilon \Delta_a^2}$  many pulls. Once exploration phase is over, the exploitation phase starts. If  $n_a(T) \leq s_a$ , then we are done, since it means the arm was never finished the exploration phase. Otherwise, we show that the probability of pulling an arm after its exploration phase is very low. In principle, what we show that for the bad arms, the expected number of pulls for those arms is bounded during its exploration phase.

Let  $X_a(t)$  be the true total reward and  $\text{Noise}_a(t) = r_a(t) - X_a(t)$  for an arm  $a$ . For the ease of notation, let  $\Gamma = \frac{k \log^2 T \log((kT \log T)/\gamma)}{\epsilon}$ . At time step  $t \in [T - 1]$ , an arm  $a$  is pulled over  $a^*$  if the following is true.

$$\begin{aligned} & \frac{r_{a^*}(t)}{n_{a^*}(t)} + \sqrt{\frac{2 \log t}{n_{a^*}(t)}} + \frac{\Gamma}{n_{a^*}(t)} \\ & \leq \frac{r_a(t)}{n_a(t)} + \sqrt{\frac{2 \log t}{n_a(t)}} + \frac{\Gamma}{n_a(t)} \\ & \Leftrightarrow \frac{X_{a^*}}{n_{a^*}(t)} + \sqrt{\frac{2 \log t}{n_{a^*}(t)}} + \frac{\Gamma}{n_{a^*}(t)} + \frac{\text{Noise}_{a^*}(t)}{n_{a^*}(t)} \\ & \leq \frac{X_a}{n_a(t)} + \sqrt{\frac{2 \log t}{n_a(t)}} + \frac{\Gamma}{n_a(t)} + \frac{\text{Noise}_a(t)}{n_a(t)} \end{aligned} \quad (2)$$

It can be easily shown that (2) is true, only if at least one of the following equations hold.

$$\frac{X_{a^*}}{n_{a^*}(t)} \leq \mu^* - \sqrt{\frac{2 \log t}{n_{a^*}(t)}} \quad (3)$$

$$\frac{X_a}{n_a(t)} \geq \mu_a + \sqrt{\frac{2 \log t}{n_a(t)}} \quad (4)$$

$$\begin{aligned} & \mu^* + \frac{\Gamma}{n_{a^*}(t)} + \frac{\text{Noise}_{a^*}(t)}{n_{a^*}(t)} \\ & < \mu_a + \frac{\Gamma}{n_a(t)} + 2\sqrt{\frac{2 \log t}{n_a(t)}} + \frac{\text{Noise}_a(t)}{n_a(t)} \end{aligned} \quad (5)$$

Directly by the use of Chernoff bound, we can show that with probability at least  $1 - 2t^{-4}$ , (3) and (4) are false. To ensure that (5) does not hold, it suffices to ensure the

following.

$$\begin{aligned} & \frac{\Gamma}{n_a(t)} + 2\sqrt{\frac{2 \log t}{n_a(t)}} + \frac{\text{Noise}_a(t)}{n_a(t)} \leq \frac{\Gamma}{n_{a^*}(t)} + \frac{\text{Noise}_{a^*}(t)}{n_{a^*}(t)} + \Delta_a \\ & \Leftrightarrow 2\sqrt{\frac{2 \log t}{n_a(t)}} + \frac{\Gamma + \text{Noise}_a(t)}{n_a(t)} - \left(\frac{\Gamma + \text{Noise}_{a^*}(t)}{n_{a^*}(t)}\right) \leq \Delta_a \end{aligned} \quad (6)$$

From Lemma 5 we know that during the execution of the algorithm, w.p.  $\geq 1 - \gamma$ ,  $|\text{Noise}_{a^*}(t)|$  and  $|\text{Noise}_a(t)|$  are at most  $\Gamma$ . Therefore, to ensure (6) it suffices to ensure the following.

$$2\sqrt{\frac{2 \log t}{n_a(t)}} + \frac{\Gamma + \text{Noise}_a(t)}{n_a(t)} \leq \Delta_a \quad (7)$$

If for some  $0 < \nu < 1$  we have,  $2\sqrt{\frac{2 \log t}{n_a(t)}} \leq \nu \Delta_a$  and  $\frac{\Gamma + \text{Noise}_a(t)}{n_a(t)} \leq (1 - \nu)\Delta_a$ , then we can claim (7). Hence

to ensure (7), by setting  $y = \sqrt{\frac{\log(t)}{\Gamma}}$  (assume  $T > 3$ ) it suffices to have  $n_a(t) \geq \frac{8\Gamma}{\Delta_a^2}$  since  $\Delta_a < 1$ . Recall that this is the exactly the threshold for the exploration phase for the arm  $a$ . We now focus our attention to the term  $\mathbb{E}[n_a(T)]$  we initially intended to bound. It is easy to see the following.

$$\begin{aligned} \mathbb{E}[n_a(T)] & \leq \left\lceil \frac{8\Gamma}{\Delta_a^2} \right\rceil + \sum_{t=\frac{8\Gamma}{\Delta_a^2}}^T \Pr[\text{Pulling arm } a \text{ at time } t] \\ & \leq \left\lceil \frac{8\Gamma}{\Delta_a^2} \right\rceil + \sum_{t=\frac{8\Gamma}{\Delta_a^2}}^T \Pr[\exists n_a(t), n_{a^*}(t) \text{ s.t. (2) holds}] \\ & \leq \left\lceil \frac{8\Gamma}{\Delta_a^2} \right\rceil + \sum_{t=\frac{8\Gamma}{\Delta_a^2}}^T \sum_{n_{a^*}=1}^t \sum_{n_a=\frac{8\Gamma}{\Delta_a^2}}^t 2t^{-4} \\ & \leq \left\lceil \frac{8\Gamma}{\Delta_a^2} \right\rceil + \sum_{t=1}^T 2t^{-2} \leq \frac{8\Gamma}{\Delta_a^2} + \zeta \end{aligned} \quad (8)$$

This completes the proof Lemma 6.  $\square$

*Proof of Theorem 4.* In order to obtain the final regret guarantee of Theorem 4 and conclude the proof, we notice that,  $\mathbb{E}[\text{Regret}_{\text{Priv-UCB}}(T)] = \mathbb{E}\left[\sum_{a \in \mathcal{C}: \mu_a < \mu_{a^*}} \Delta_a n_a(T)\right]$ . Using Lemma 6 in the expression above concludes the proof.  $\square$

## 4 PRIVATE THOMPSON SAMPLING

In this section we study a different flavor of bandit learning algorithms called *Thompson sampling*. Historically,

Thompson sampling [25] is much older than UCB sampling style algorithms, dating back to early 20<sup>th</sup> century. Although Thompson sampling is a very powerful heuristic and often outperforms UCB sampling in experimental setups, until recently little was known about its regret guarantees. [2] provided the first regret analysis for Thompson sampling and they showed that it has regret logarithmic in the time horizon  $T$ . We provide a differentially private variant of the Thompson sampling algorithm. One reason for studying private Thompson sampling along with private UCB sampling is because it is known that in the non-private world the experimental performance of Thompson sampling is much better than UCB sampling. Moreover it demonstrates properties like stability to delayed feedback. (See [8] for details.) The question we want to answer is whether we can obtain a differentially private variant of Thompson sampling which preserves the asymptotic regret guarantee of as the same order as the non-private algorithm and also demonstrate similar experimental properties.

**Background on Thompson sampling.** The basic Thompson sampling heuristic is extremely simple. Suppose there are  $k$ -arms  $\mathcal{C} = \{a_1, \dots, a_k\}$  which give Bernoulli rewards (i.e.,  $\{0,1\}$  rewards) and have biases  $\mu_{a_1}, \dots, \mu_{a_k}$ . Let  $r_{a_1}(t), \dots, r_{a_k}(t)$  be the number of ones and  $n_{a_1}(t), \dots, n_{a_k}(t)$  be the total number of arm pulls, upto time  $t$ . The rule to pick the arm for each round is: Sample  $\theta_i \sim \text{Beta}(r_{a_i}(t) + 1, n_{a_i}(t) - r_{a_i}(t) + 1)$  for  $i \in [k]$  for  $i \in [k]$  and pull the arm corresponding to the highest  $\theta_i$ . Based on the result of the arm pull, the posterior distribution for that particular arm gets updated.[2] showed that the expected regret for this algorithm (over the randomness of the algorithm and the data) is  $O\left(\sum_{a \in \mathcal{C} - a^*} \left(\frac{1}{\Delta_a^*}\right)^2 \log T\right)$ , where  $a^*$  be the optimal arm and  $\Delta_a = \mu_{a^*} - \mu_a$ .

#### 4.1 Private Thompson Sampling: Algorithm and Analysis

In this section we modify this generic Thompson sampling algorithm to obtain an  $\epsilon$ -differentially private variant (Algorithm 2). We show that even with the privacy constraint our algorithm has almost the same regret as non-private algorithm with only poly  $\log T$  overhead. This is the same setting in which [2] also proved their results. One interesting observation in the MAB problems is that all the sequential algorithms would encourage a downward bias, which means that if an arm does not give good results initially then it will not be pulled again, therefore it's empirical mean would be much lower than its true mean. UCB algorithm overcomes this problem by adding a monotonically decreasing function of the number of pulls ( $n_a(t)$ ) to the empirical mean, thus in some sense balancing out this downward bias. But in case of Thompson sampling, we are randomizing our decision hence the bias correction mech-

anism is different, it is due to randomization. In Thompson sampling, the biggest challenge is to bound the number of mistakes in the initial rounds. Moreover to ensure differential privacy, we introduce additional randomness to the original Thompson sampling algorithm, it becomes even harder for us to analyze the number of mistakes in the initial rounds. So, we take a slightly different approach. We segregate the algorithm into *explicit exploration phase* and a *combined exploration and exploitation* phase. The idea in the exploration phase is to estimate the biases of the two arms (within sufficient confidence) without bothering about the number of mistakes made. In the joint exploration and exploitation phase, we use the standard Thompson sampling, except we make sure the algorithm only has differentially private access to the rewards. Similar to the private UCB algorithm (Algorithm 1), we use the *private tree based aggregation* (from Section 2.1) to ensure differential privacy. In the following section, for the convenience of notation we assume that  $\mu^* = \mu_{a_1} > \mu_{a_2} \geq \dots \geq \mu_{a_k}$ .

##### 4.1.1 Privacy Analysis

**Theorem 7** (Privacy guarantee). *Algorithm 2 is  $\epsilon$ -differentially private.*

*Proof.* The proof of privacy is segregated into two parts. In the first part we argue that the *gap estimation* section is  $\epsilon/2$ -differentially private. In the second part we argue that the rest of the sections of the algorithm are combined  $\epsilon/2$ -differentially private. Using these two arguments and the composition property of differential privacy we argue that Algorithm 2 is  $\epsilon$ -differentially private. Notice that in the first part, each arm  $i, i \in [k]$  is pulled in batches of  $m$ -pulls, till the condition in Line 8 in Algorithm 2 is satisfied. If each of this batch is made  $\frac{\epsilon}{2k}$ -differentially private, then by *parallel composition* property of differential privacy, the first phase is  $\epsilon/2$ -differentially private. To guarantee  $\frac{\epsilon}{2k}$ -differential privacy for a given batch, we use Laplace mechanism from Section 2.1. The  $\epsilon/2$ -privacy guarantee for the second phase follows directly from the analysis of tree based aggregation scheme described in Section 2.1.  $\square$

##### 4.1.2 Regret Analysis

In this section, we provide the regret analysis for our private Thompson sampling algorithm (Algorithm 2). The high-level structure of the analysis is as follows. First we establish that in the gap estimation phase, the estimated gap  $\hat{\Delta}$  is within constant factor of the true gap  $\Delta = |\mu_{a_{(1)}} - \mu_{a_{(2)}}|$  (i.e. the mean difference of the best arm and the second best arm). Moreover, we argue that the gap estimation runs for at most poly  $\log T$  number of rounds. Second, assuming our estimation  $\hat{\Delta}$  is reasonably accurate, we pull all the arms randomly for certain number of steps to ensure sufficient concentration of the empirical means around the true means. In the third and final stage, we analyze the noisy version of the classic Thompson sampling with *beta* prior. The analysis of this part resembles the analysis of [2]

---

**Algorithm 2** Differentially Private Thompson Sampling

- Input:** Time horizon:  $T$ , arms:  $\mathcal{C} = \{a_1, a_2, \dots, a_k\}$ , privacy parameter:  $\epsilon$ .
- 1: **Gap** ( $\Delta = |\mu_{a_1} - \mu_{a_2}|$ ) **estimation**
  - 2: **Initialize:** Time counter:  $\tau \leftarrow 0$ , estimated gap:  $\hat{\Delta} \leftarrow 1$ , total pulls:  $n_{a_1}, n_{a_2}, \dots, n_{a_k}$ .
  - 3: **repeat**
  - 4: Pull each arm  $a_i$   $m = \frac{192k \log T}{\epsilon \hat{\Delta}^2}$  times to obtain average rewards  $\tilde{\mu}_{a_i} \forall a_i \in \mathcal{C}$ . Increment  $n_{a_i} \leftarrow n_{a_i} + m \forall a_i \in \mathcal{C}$ .
  - 5: **Differentially private means:**  $\hat{\mu}_{a_i} \leftarrow \tilde{\mu}_{a_i} + \text{Lap}\left(\frac{2k}{\epsilon m}\right), \forall a_i \in \mathcal{C}$ .
  - 6: Set  $\hat{\Delta} \leftarrow \hat{\Delta}/2$  and  $\tau \leftarrow \tau + km$ .
  - 7: Find the best and second best arms as,  $a(1) = \operatorname{argmax}_{a_i \in \mathcal{C}} \mu_{a_i}$  and  $a(2) = \operatorname{argmax}_{a_i \in \mathcal{C}, a_i \neq a(1)} \mu_{a_i}$
  - 8: **until**  $|\hat{\mu}_{a(1)} - \hat{\mu}_{a(2)}| > \hat{\Delta}$
  - 9: Create empty trees  $\text{Tree}_{a_i}$  with  $(T - \tau)$ -leaves  $\forall a_i \in \mathcal{C}$ . Set  $\epsilon_0 \leftarrow \epsilon/2k$ .
  - 10: **Random pullings of arms to build confidence**
  - 11: **Confidence parameter:**  $\Gamma \leftarrow \frac{192 \log^3 T}{\epsilon}$ .
  - 12: Pull each arm  $a_i$ ,  $\frac{\Gamma}{\hat{\Delta}^2}$  times. Record the total rewards  $r_i \forall a_i \in \mathcal{C}$ . Insert  $r_i$  in  $\text{Tree}_{a_i}$  with privacy parameter  $\epsilon_0, \forall a_i \in \mathcal{C}$ . Increment  $n_{a_i} \leftarrow n_{a_i} + \frac{\Gamma}{\hat{\Delta}^2}$  and  $\tau \leftarrow \tau + \frac{k\Gamma}{\hat{\Delta}^2}$ .
  - 13: **Combined explore-exploit phase of Thompson sampling**
  - 14: **for**  $t \leftarrow \tau + 1$  to  $T$  **do**
  - 15: **Total reward:**  $r_a(t) \leftarrow$  Total reward computed using  $\text{Tree}_{a_i}, \forall a_i \in \mathcal{C}$ .  $\theta_a(t) \sim \text{Beta}(r_a(t) + 1, n_a(t) - r_a(t) + 1), \forall a_i \in \mathcal{C}$ . Pull arm  $a^* = \operatorname{argmax}_{a_i \in \mathcal{C}} (\theta_a(t))$  and observe reward  $f_t(a^*)$ , Insert  $f_t(a^*)$  into  $\text{Tree}_{a^*}$  using *tree based aggregation* and privacy parameter  $\epsilon_0$ ,
  - 16: **Number of pulls:**  $n_{a^*}(t) \leftarrow n_{a^*}(t) + 1$ .
  - 17: **end for**
- 

closely. The overall regret guarantee is given in Theorem 8 below.

**Theorem 8** (Utility guarantee). *Let  $\mu_{a_1}, \mu_{a_2}, \dots, \mu_{a_k}$  be the biases of the  $k$  arms. Let  $\Delta = |\mu_{a(1)} - \mu_{a(2)}|$ , where  $a(1) = \operatorname{argmax}_{a_i \in \mathcal{C}} \mu_{a_i}$  and  $a(2) = \operatorname{argmax}_{a_i \in \mathcal{C}, a_i \neq a(1)} \mu_{a_i}$ .*

*Then, for  $T \geq 8 \max\{\log_2 \frac{1}{\Delta}, 1\}$  and  $\epsilon < 1$  expected regret of Algorithm 2 (over the randomness of the data and the algorithm) is as follows:*

$$\mathbb{E} [\text{Regret}_{\text{Priv-Thompson}}(T)] = O\left(k \frac{\log^3 T}{\Delta^2 \epsilon^2}\right)$$

Let  $N_1, N_2$  and  $N_3$  denote the number of times the wrong arm is pulled in the *gap estimation phase*, *random pullings phase* and *combined explore and exploit phase* of Algorithm 2 respectively. We would bound the expected values of  $N_1, N_2$  and  $N_3$  using the following lemmas which would allow us to prove Theorem 8.

**Lemma 9** (Bound on gap estimation phase). *Following the notation of Theorem 8, then with probability at least  $1 - 1/T^4$  (over the randomness of the algorithm and the data distribution), for  $T \geq 8k \max\{\log_2 \frac{1}{\Delta}, 1\}$  and  $\epsilon < 1$  the estimated gap  $\hat{\Delta}$  in Algorithm 2 is in  $[\Delta/3, 2\Delta]$ , the expected number of times incorrect arm is pulled is,  $\mathbb{E}[N_1] = O\left(\frac{k \log_2 T}{\epsilon}\right)$ .*

The proof of this Lemma is provided in Appendix C.

**Lemma 10** (Bound on geometric random variables (Lemma 3 [2])). *Let  $s_j$  be a random variable, index by  $j \in \mathbb{Z}^+$  s.t. for fixed parameters  $T \in \mathbb{Z}^+$  and  $\mu \in [0, 1]$ , with probability at least  $1 - T^{-2}$ ,  $\frac{s_j}{j} > \frac{\mu + y}{2}$ . Let  $y \in [0, \mu]$  be a predefined threshold and let  $X(s_j, y)$  be the random variable that counts the number samples  $w \sim \text{Beta}(s_j + 1, j - (s_j + 1))$  that need to be drawn i.i.d. before  $w$  exceeds  $y$ . Using  $T$  as the fixed upper bound on  $X(s_j, y)$ , if  $j \geq \frac{4 \log_2 T}{(\mu - y)^2}$ , then  $\mathbb{E}_s [\mathbb{E}_w [\min\{X(s_j, y), T\}]] = O\left(\frac{1}{T}\right)$ .*

**Lemma 11** (Bound on Beta random variable [Lemma 7 [2]]). *Let us define event  $E(t)$  as,  $E(t) : \theta_a(t) \in [\mu_a - \frac{\Delta_a}{2}, \mu_a + \frac{\Delta_a}{2}], \forall a \in \mathcal{C}, a \neq a(1)$ . Then  $\Pr(E(t), n_a(t) \geq \frac{32 \log_2 T}{\Delta^2}) \geq 1 - \frac{4(k-1)}{T^3}, \forall t$ .*

**Lemma 12** (Regret bound in the combined explore and exploit phase). *Suppose  $N_3$  is a random variable which counts the number of times the suboptimal arms are pulled in Algorithm 2. Then, over the randomness of the algorithm and the data,  $\mathbb{E}[N_3] = O(1)$ .*

*Proof.* In order to calculate the regret we count the number of times the sub-optimal arms get pulled since it upper bounds the regret. The main idea behind the proof of this lemma is that via random pulling of the arms in Algorithm 2, the arm  $a_1$  is well separated from the other sub-optimal arms. Hence, in the combined explore and exploit phase, with high probability the optimal arm gets pulled almost always. We will use the proof technique from [2] for analyzing the number of pulls of the suboptimal arms.

We denote the set of suboptimal arms by  $\mathbb{S}$  and  $\mathbb{S} = a_2, \dots, a_k$ . We count the number of pulling of sub-optimal arm  $a_s$ , by the following scheme: count the number of pulls of arm  $a_s$  in between two successive pulls of arm  $a_1$ . First, recall that with probability at least  $1 - T^{-4}$  (from Lemma 9),  $\hat{\Delta}$  (the estimated gap) is within  $[\Delta/3, 2\Delta]$ . By the property of the tree based aggregation discussed earlier, the difference between the true total reward at time  $t(j)$  and  $r_a(t(j))$  is at most  $\frac{\log_2^3 T}{\epsilon}$ , with probability  $1 - \text{negl}(T)$ . Since by the beginning of combined explore and exploit phase, we have pulled arm  $a_1$  at least  $\frac{\Gamma}{\hat{\Delta}^2}$ -times, it follows that with probability at least  $1 - 2T^{-4}$ , difference between the true total reward at time  $t(j)$  and  $r_{a_1}(t(j))$  is at most  $\frac{\Delta}{4}$ . Using Lemma 11 and the fact that we pulled all the arms for  $\Gamma/\Delta^2 > \frac{32 \log_2 T}{\Delta^2}$ , we observe that with probability atleast



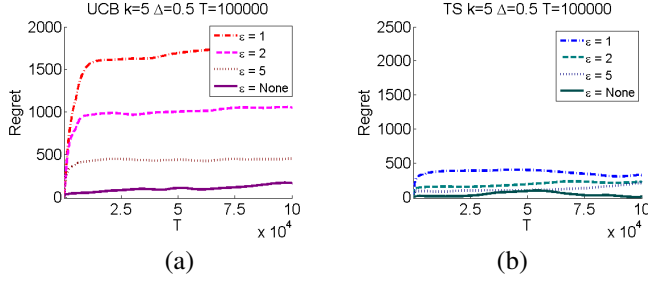


Figure 1: Simulation results for our differentially private algorithms UCB sampling (Algorithm 1) and Thompson sampling (Algorithm 2) for the number of arms  $k = 5$  and  $\Delta = 0.5$ . Smaller  $\epsilon$  indicates more privacy.

$1 - \frac{4k-2}{T^3}$ ,  $\theta_a < \mu_a + \Delta_a/2$ . Hence bundled up failure probabilities obtained so far for  $\theta_{a_1} < \theta_{a_s}$  using union bound is  $1 - \frac{4}{T^2}$ . Let  $j$  and  $j+1$  be any two successive pulls of arm  $a_1$ , let  $t(j)$  and  $t(j+1)$  be the respective time epochs and denote  $\Delta_s = \mu_{a_s} - \mu_{a_1}$ . The Beta distribution for the sample  $\theta_a$  from arm  $a$  in between these pulls (not including the  $(j+1)$ -th pull) is  $\text{Beta}(r_a(t(j))+1, n_a(t(j))-r_a(t(j))+1)$ . Now a suboptimal arm  $a_s$  is pulled during round  $j$  and  $j+1$  pull of arm  $a_1$  if  $\theta_{a_1} < \mu_{a_1} - \min_{a_s \in \mathbb{S}} \Delta_{a_s}$ . In order to use Lemma 10 above, we set the threshold  $y = \mu_{a_2} + \frac{\Delta}{2}$ . Moreover, if we set  $s_j = r_{a_1}(t(j))$ , then we know that with probability at least  $1 - 4T^{-2}$ ,  $\frac{s_j}{j} > \mu_{a_1} - \frac{\Delta}{4}$ . Hence, the conditions of Lemma 10 holds and summing over all  $T$  rounds, we complete the proof.  $\square$

*Proof of Theorem 8.* The expected regret of multi-armed bandit algorithms can be upper bounded by the number of times the suboptimal arm is pulled. We defined  $N_1$ ,  $N_2$  and  $N_3$  to denote the number of times the wrong arm is pulled in the *gap estimation* phase, *random pullings* phase of Algorithm 2 and *combined explore and exploit* phase of Algorithm 2 respectively. Thus we have,

$$\mathbb{E}[\text{Regret}_{\text{Priv-Thompson}}(T)] = \mathbb{E}[N_1] + \mathbb{E}[N_2] + \mathbb{E}[N_3]$$

From Lemma 9 we know that  $\mathbb{E}[N_1] = O\left(k \frac{\log T}{\epsilon^2}\right)$ . From Lemma 9 and Algorithm 2, we know that  $\mathbb{E}[N_2] = O\left(k \frac{\Gamma}{\Delta^2}\right)$ , where  $\Gamma = O\left(\frac{\log^3 T}{\epsilon}\right)$ . Finally from Lemma 12, we know that  $\mathbb{E}[N_3] = O(1)$ . Combining these bounds, we get the bound on the expected regret.  $\square$

## 5 EXPERIMENTAL EVALUATION

In this section, we support the theoretical regret bounds for our algorithms (Algorithm 1 and 2) with empirical results. The experimental results show that there is a smooth trade-off between privacy and accuracy. As we increase our privacy parameter  $\epsilon$ , the regret improves. We perform the simulation experiments on for stochastic multi-arm bandits, with rewards in  $\{0, 1\}$ . The  $k$ -arm private UCB sampling algorithm is described in Section 3. The 2-arm private Thompson sampling and its extension to  $k$ -arm private

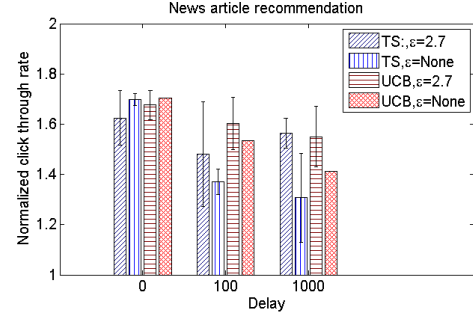


Figure 2: Comparison of differentially private and non-private multi-armed bandit algorithms on Yahoo! front page News article recommender system. The click-through rates for each algorithm is normalized with respect a random algorithm.

Thompson sampling are given in Section 4.1. The true underlying distribution of the arms are chosen as follows. The mean for the best arm is 0.9 and the other arms have biases of  $0.9 - \Delta$  each, where  $\Delta = 0.5$ . We tune the parameters of our private algorithms, the *confidence parameter* (see Line 9 in Algorithm 1) for UCB and the number of *random pullings* (see Line 12 in Algorithm 1) for Thompson to enhance accuracy. Note, the parameter tuning does not violate privacy, since the access to the aggregated rewards is still based the on tree based aggregation scheme. All the UCB type sampling algorithms have a common parameter, the confidence interval; same as Line 9 in Algorithm 1. For our experiment on private UCB sampling (Algorithm 1), we use a particular confidence interval, given in [8], which seems to perform the best. The confidence interval is given as,  $\frac{\sqrt{r_a(t) \log t}}{n_a(t)} + \frac{\log t}{n_a(t)}$ , where  $r_a(t)$  and  $n_a(t)$  are the reward and number of pulls for arm  $a \in \mathcal{C}$  up to time  $t$  respectively. For our experiments on private Thompson sampling (see Algorithm 2) we do not implement the *gap estimation* phase and for the second phase that involves *random pullings* (Line 10 of 2), use a smaller value for  $m$ .

**Conclusions drawn from simulations.** We observe in the plots that the regret for the private algorithms saturates after certain time, similar to that of their non-private counterparts (see Figure 1). Also notice that in the simulations, Thompson sampling tends to perform much better than UCB sampling.

### 5.1 Yahoo! Front Page Data set

In this section, we describe our results on *Yahoo! front page news article recommendation* data set. The data set contains 45,811,883 user visits to the *Today module* during first 10 days in May 2009. Each user click on a news article shown corresponds to a reward of one for that article. This data set has also been used by [21], [11] for bandit experiments. One property of this data set is that the displayed article is chosen uniformly at random from the candidate article pool allowing us to use an *unbiased offline* evalua-

tion method [21, 22]. The pool of articles is small (around 20 articles), but it is dynamic which means that the articles may be added or removed from this pool. For each visit, both the user and each of the candidate articles are associated with a feature vector of dimension 6. The feature vector acts as a context for the news article recommender and based on this context the most suited article can be chosen using a bandit algorithm. This is the contextual bandit setting. In this setting, in each of  $T$  rounds, a learner is presented with the context vector:  $z_a \in \mathbb{R}^d$  for each arm  $a \in \mathcal{C}$  and based on his previous observations and this new context vector, the learner needs to select one out of  $k$  actions. The learner’s aim is to learn the relation between the reward and the context vector in an online fashion.

***Differentially private contextual sampling (Algorithm 3 and Algorithm 4 in Appendix D.2 and D.3.)*** The private contextual UCB algorithm is adapted from the *LinUCB* algorithm in [21] and is similar to the basic UCB sampling algorithm, as it computes the expected reward of each arm and then chooses the arm with the highest upper confidence bound. The expected reward is given as  $z_t^T \theta_t$ , where  $\theta_t$  is estimated using ridge regression and the confidence bound is given as  $\sqrt{z_a(t)^T A_t z_a(t)}$ , which is the Mahalanobis distance of the context vector with covariance matrix  $A$ . On the other hand, the private contextual Thompson sampling algorithm is a differentially private version for the algorithm provided by [3]. In regular Thompson sampling, for each round we choose an arm according to its posterior probability of having the best parameter. A natural generalization of Thompson Sampling for contextual bandits is to use Gaussian prior and Gaussian likelihood function. The extension of these algorithms to private algorithms is straightforward. In both the private algorithms, we restrict our access to the parameters which aggregate over each time stamp and use *tree based aggregation* scheme to retrieve those parameters. We give the details of these algorithms in Appendix D.2 and D.3.

***Conclusions on experiments with Yahoo! front page data set.*** The results for this experiment are summarized in Figure 2. We find that the private algorithms do not perform much worse than the non-private algorithms. Since, the feature vector is of length 6 by setting the privacy  $\epsilon_0$  of each parameter as 0.1, the total privacy measured in terms of the total privacy parameter  $\epsilon = 2.7$ . We also investigate the performance of the algorithms with respect to delays. We have considered the delay values in  $\{0, 100, 1000\}$ . When the input data does not have any delay in the feedback, the private algorithms perform slightly worse than the non-private counter parts and as the delay increases the performance of the non-private algorithms is hurt more than the private algorithms.

## References

- [1] Alekh Agarwal, Ofer Dekel, and Lin Xiao. Optimal algorithms for online convex optimization with multi-point bandit feedback. In *COLT*, pages 28–40, 2010.
- [2] Shipra Agrawal and Navin Goyal. Analysis of thompson sampling for the multi-armed bandit problem. In *COLT*, 2012.
- [3] Shipra Agrawal and Navin Goyal. Thompson sampling for contextual bandits with linear payoffs. *arXiv preprint arXiv:1209.3352*, pages 1–29, 2012.
- [4] Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.
- [5] Sébastien Bubeck and Nicolò Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *CoRR*, abs/1204.5721, 2012.
- [6] Joseph A. Calandrino, Ann Kilzer, Arvind Narayanan, Edward W. Felten, and Vitaly Shmatikov. ”you might also like: ” privacy risks of collaborative filtering. In *IEEE Symposium on Security and Privacy*, 2011.
- [7] TH Hubert Chan, Elaine Shi, and Dawn Song. Private and continual release of statistics. In *ICALP*. 2010.
- [8] Olivier Chapelle and Lihong Li. An empirical evaluation of thompson sampling. In *Advances in Neural Information Processing Systems*, pages 2249–2257, 2011.
- [9] Kamalika Chaudhuri. Topics in online learning: Lecture notes. 2011.
- [10] Kamalika Chaudhuri and Claire Monteleoni. Privacy-preserving logistic regression. In Daphne Koller, Dale Schuurmans, Yoshua Bengio, and Léon Bottou, editors, *NIPS*. MIT Press, 2008.
- [11] Wei Chu, Seung-Taek Park, Todd Beaupre, Nitin Motgi, Amit Phadke, Seinjuti Chakraborty, and Joe Zachariah. A case study of behavior-driven conjoint analysis on yahoo!: front page today module. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1097–1104. ACM, 2009.
- [12] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography Conference*, pages 265–284. Springer, 2006.
- [13] Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N Rothblum. Differential privacy under continual observation. In *STOC*, 2010.
- [14] Cynthia Dwork, Moni Naor, Omer Reingold, Guy Rothblum, and Salil Vadhan. On the complexity of differentially private data release: efficient algorithms and hardness results. In *STOC*, pages 381–390, 2009.
- [15] Abraham D Flaxman, Adam Tauman Kalai, and H Brendan McMahan. Online convex optimization in the bandit setting: gradient descent without a gradient. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 385–394. Society for Industrial and Applied Mathematics, 2005.
- [16] Prateek Jain, Pravesh Kothari, and Abhradeep Thakurta. Differentially private online learning. *arXiv preprint arXiv:1109.0105*, 2011.
- [17] Prateek Jain, Pravesh Kothari, and Abhradeep Thakurta. Differentially private online learning. In *Conference on Learning Theory*, pages 24.1–24.34, 2012.
- [18] Emilie Kaufmann, Nathaniel Korda, and Rémi Munos. Thompson sampling: An asymptotically optimal finite-time analysis. In *Algorithmic Learning Theory*, pages 199–213. Springer, 2012.
- [19] Aleksandra Korolova. Privacy violations using microtargeted ads: A case study. In *International Conference on Data Mining Workshops*, 2010.
- [20] Tze Leung Lai and Herbert Robbins. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, 6(1):4–22, 1985.
- [21] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670. ACM, 2010.
- [22] Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 297–306. ACM, 2011.
- [23] Shai Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2011.
- [24] Adam Smith and Abhradeep Thakurta. Nearly optimal algorithms for private online learning in full-information and bandit settings. In *NIPS (To appear)*, 2013.
- [25] William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.

---

# Equitable Partitions of Concave Free Energies

---

**Martin Mladenov**  
TU Dortmund University  
{fn.in}@cs.tu-dortmund.de

**Kristian Kersting**  
TU Dortmund University  
{fn.in}@cs.tu-dortmund.de

## Abstract

Significant progress has recently been made towards formalizing symmetry-aware variational inference approaches into a coherent framework. With the exception of TRW for marginal inference, however, this framework resulted in approximate MAP algorithms only, based on equitable and orbit partitions of the graphical model. Here, we deepen our understanding of it for marginal inference. We show that a large class of concave free energies admits equitable partitions, of which orbit partitions are a special case, that can be exploited for lifting. Although already interesting on its own, we go one step further. We demonstrate that concave free energies of pairwise models can be reparametrized so that existing convergent algorithms for lifted marginal inference can be used without modification.

## 1 INTRODUCTION

Computing likelihoods and marginals using graphical models [24] is an important task for many applications in biology, information retrieval, and computer vision, among other fields. If the graphical models are defined over trees, marginals can be efficiently computed using belief propagation. For models with cycles, however, exact inference is generally intractable. This motivates approximate inference algorithms, favoring algorithms which are as accurate as possible while being guaranteed to converge. One prominent example are variational inference approaches [24, 7, 14], where one aims to approximate a given distribution by a simpler one, i.e., one whose marginals are easier to read off. If a good fit is found, the marginals of the approximating distribution can be used as approximations to the marginals of the original one. Such approaches are typically obtained in two steps: (1) one selects an approximation criterion (a free energy), which is a function of the

approximating marginals, and (2) designs optimization algorithms to minimize that free energy efficiently.

A recent development in probabilistic inference has been the use of symmetry [18, 1, 19, 23] as a basis for efficient algorithms. Detecting and utilizing symmetry is establishing itself as an important component of inference. On one hand, there are classes of models where symmetry provides the only means for tractable inference [4]. On the other, in approximate inference algorithms (which tend to be tractable by design) symmetry usually translates to significant improvements in running time as a result of reducing the number of variables of the problem. Symmetry-aware inference approaches are often referred to as lifted inference approaches [20, 9], and one of the first lifted variational inference approaches was lifted loopy belief propagation [8, 22, 10]. These works, however, are largely of algorithmic nature and specific to loopy belief propagation. More recently, Bui et al. [1] proposed a general, algebraic framework for lifted variational inference. It formalizes the notion of symmetry in graphical models via lifting partitions and shows how to exploit them within corresponding variational optimization problems. With the exception of lifted TRW via Frank-Wolfe optimization [2], however, the framework resulted in approximate MAP inference algorithms only, based on equitable and orbit partitions of the graphical model [1, 17, 16].

Our goal in this paper is to deepen our understanding of the lifted variational framework for marginal inference. We do so by extending the notion of equitable partitions — a formalization of symmetry — of models to equitable partitions of energies. We show that within a well-known class of concave energies, **a**) given a concave energy that admits an equitable partition, the number of variables in the resulting optimization problem can be reduced in a way that an exact solution can still be found. Moreover, **b**) given a model that admits an equitable partition, a concave energy that admits the same equitable partition can always be constructed. In combination, these two results allow us to perform concave inference without breaking the symmetry of the model. Although already interesting on its

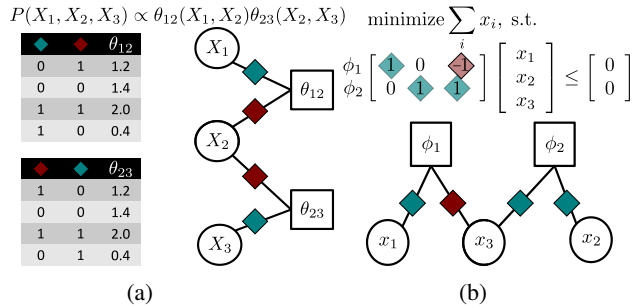


Figure 1: Representations. (a) An MRF and its factor graph. (b) An LP and its factor graph. (Best viewed in color)

own, we go one step further. We demonstrate that for the case of pairwise models, “lifted” concave free energies can be reparametrized to “ground” energies of smaller models. That is, for any pair  $(G, c)$  of pairwise model and energy parameters as well as an appropriate partition, we can find a pair  $(G', c')$  of smaller size such that the resulting energies are equivalent, regardless of the solver. This enables us to use existing highly efficient and distributed convergent message passing algorithms for lifted marginal inference such as ([21]) without modification. Furthermore, we provide a novel angle on the open question raised by Bui et al. regarding the applicability equitable partitions within lifted variational inference. While they point out that partitions coarser than an orbit partition of the model generally cannot lift the TRW energy [2] faithfully, we show that a large class of other energies in the same class do admit any equitable partition of the model.

To achieve the above goal(s) we start off by reviewing the required tools from (lifted) variational inference. Section 3 then introduces the notion of an equitable partition of a convex energy and shows that this partition is a lifting partition. In Section 4, we show to reparametrize energies and models modulo equitable partitions in order to eliminate the necessity of a lifted solver. Before concluding, we illustrate our theoretical results empirically.

## 2 BACKGROUND

We will start with reviewing variational inference in Markov random fields (MRF). Then we will touch upon the basics of the lifting framework for variational problems.

**Variational Inference in MRFs.** Let  $\mathbf{X} = (X_1, X_2, \dots, X_n)$  be a set of  $n$  discrete-valued random variables and let  $x_\alpha$  represent the possible realizations of a subset  $\alpha$  of these random variables. Markov random fields (MRFs) compactly represent a joint distribution over  $\mathbf{X}$  as a factorization  $P(\mathbf{X} = \mathbf{x}) = Z^{-1} \exp[\sum_\alpha \theta_\alpha(x_\alpha) + \sum_i \theta_i(x_i)]$ , see [24] for more details.

It is often convenient to represent MRFs by their factor

graphs. In this paper, however, we will slightly modify the standard definition of a factor graph. For our purposes, a **factor graph**  $G$  is a colored tri-partite graph, whose nodes represent the variables, factors and the positions of variables in factors within an MRF. In contrast to standard factor graphs and as illustrated in Fig. 1a, we connect a variable  $X_i$  to factor  $\theta_\alpha$  via a dummy position node  $\diamond_{i\alpha}$ , which we color according to the symmetry of  $\theta_\alpha$ . More precisely, if the positions of  $X_i$  and  $X_j$  are compatible, that is,  $\theta_\alpha(\dots, x_i, \dots, x_j, \dots) = \theta_\alpha(\dots, x_j, \dots, x_i, \dots)$  for all realizations  $x_i, x_j$ , we color  $\diamond_{i\alpha}$  and  $\diamond_{j\alpha}$  with the same color. If the positions are not compatible, they receive different colors. Moreover, we assume that factors, variables and positions use different color spaces, e.g. a position and factor node cannot share the same color. While this is not the most compact representation, it will allow us to use a common graphical representation across various kinds of partitions and optimization problems.

Inference in MRFs is generally intractable, hence, inference tasks are often addressed via approximations. One prominent class of approximate inference algorithms arises from the following optimization problem:

$$\boldsymbol{\mu}^* = \operatorname{argmax}_{\boldsymbol{\mu} \in \mathcal{L}(G)} \underbrace{\left[ \boldsymbol{\theta}^T \boldsymbol{\mu} + T \cdot \hat{H}(\boldsymbol{\mu}) \right]}_{=: F(\boldsymbol{\mu})}, \quad (1)$$

where  $F$  is the free energy and the set  $\mathcal{L}(G)$ , defined as

$$\mathcal{L}(G) = \left\{ \boldsymbol{\mu} \geq 0 \mid \begin{array}{l} \sum_{x_i} \mu_i(x_i) = 1 \\ \sum_{x_\alpha \setminus x_i} \mu_\alpha(x_\alpha) = \mu_i(x_i) \end{array} \right\}, \quad (2)$$

is known as the local polytope [24]. The problem in Eq. 1 is at the heart of many message-passing inference algorithms. For instance, if we set  $T = 0$  (or sufficiently small in the sense of [13]),  $\boldsymbol{\mu}^*$  in Eq. 1 yields a linear programming approximation of the Maximum a-Posteriori (MAP) problem and prominent MAP algorithms such as MPLP and MSD are typically derived as specialized solvers for the latter. If, on the other hand, we choose  $T$  to be 1 and  $\hat{H}$  to be an approximation of the entropy function,  $\boldsymbol{\mu}^*$  approximates the vector of single-node and factor marginals of the distribution  $P$ . For example, we can choose  $\hat{H} = \hat{H}_c$  as

$$\hat{H}_c(\boldsymbol{\mu}) = \sum_i c_i H_i(\mu_i) + \sum_\alpha c_\alpha H_\alpha(\mu_\alpha), \quad (3)$$

where  $H_i$  and  $H_\alpha$  are local entropies. For  $c_i = 1 - |\text{nb}(i)|$  and  $c_\alpha = 1$ ,  $F$  becomes the Bethe energy,  $F_{\text{Bethe}}$ . In this case, solving the set of saddle-point equations of Eq. 1 by means of fixed-point iteration yields the popular Loopy Belief Propagation algorithm. The Bethe Energy often gives surprisingly good approximations to the true marginals, however, it is rather difficult to optimize over. Thus, one may prefer to consider instances of  $\hat{H}_c$ , where maximization is efficient.

Naturally, a class of such energies results from  $\hat{H}$  being concave. In particular, we are interested in values of  $c$  that

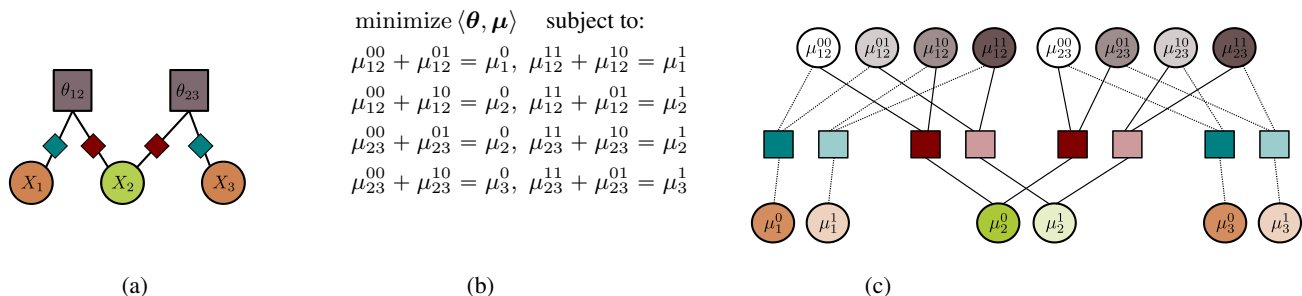


Figure 2: Model symmetry of  $G$  propagates to its MAP-LP. (a)  $G$  – colored by the classes of the CEP. (b) The MAP-LP of  $G$  (nonnegativity and normalization constraints as well as position nodes have been omitted for clarity). (c) The factor graph of the MAP-LP of  $G$  colored by its CEP. The colors indicate that the classes can be deduced from the CEP of  $G$ . Note that the darker and lighter version of each color are not grouped together. (Best viewed in color)

make  $\hat{H}_c$  in Eq. 3 concave, as the structure of these energies gives rise to message-passing algorithms with desirable theoretical properties, cf. [14, 6]. A sufficient condition for  $\hat{H}_c$  to be concave is that nonnegative auxiliary numbers  $c_{\alpha\alpha}$ ,  $c_{ii}$ , and  $c_{i\alpha}$  exist, called **counting numbers**, obeying to

$$C(G) = \left\{ c_\alpha, c_i \mid \begin{array}{l} \exists c_{\alpha\alpha}, c_{ii}, c_{i\alpha} \geq 0, \\ c_\alpha = c_{\alpha\alpha} + \sum_{i \in \alpha} c_{i\alpha}, \\ c_i = c_{ii} - \sum_{\alpha: i \in \alpha} c_{i\alpha} \end{array} \right\}. \quad (4)$$

Finally, note that for  $T = 0$ , Eq. 1 becomes a linear program (LP), called the MAP-LP of  $G$ . An LP is an optimization problem of the form maximize  $c^T x$  subject to  $Ax \leq b$ . While LPs are not the focus of this paper, they will be an important tool in the analysis of variational inference problems. Note that linear programs, like MRFs, can be represented by factor graphs. We represent a constraint  $\alpha$  (row  $\alpha$  of  $A$ ) as a factor node  $\phi_\alpha$ , LP variable  $i$  as a variable node  $x_i$  and the coefficient  $A_{\alpha i}$  as position node  $\diamond_{\alpha i}$ . Our notion of compatibility of positions here is that  $\diamond_{\alpha i}$  is colored with the same color as  $\diamond_{\beta j}$  if  $A_{\alpha i} = A_{\beta j}$ . Additionally,  $x_i$  and  $x_j$  are colored with the same color if  $c_i = c_j$ , while  $\phi_\alpha$  and  $\phi_\beta$  are colored the same if  $b_\alpha = b_\beta$ . An example is given in Fig. 1b. Having set-up the variational inference problem, we now give a short review of lifted variational inference.

**Lifted Variational Inference via Lifting Partitions.** Lifted inference approaches essentially amount to reducing the size a model by grouping together indistinguishable variables and factors. In other words, they exploit symmetries. To formalize the notion of symmetry more concisely, we follow [1].

Consider the linearly constrained concave program

$$x^* = \operatorname{argmax}_{Ax \leq b} J(x). \quad (\clubsuit)$$

We are interested in partitioning the variables of the program by a partition  $\mathcal{P} = \{P_1, \dots, P_p\}$ ,  $P_i \cap P_j = \emptyset$ ,

$\bigcup_i P_i = [x_1, \dots, x_n]$ , such that there exists at least one solution that **respects** the partition. More formally,  $\mathcal{P}$  is a **lifting partition** of  $(\clubsuit)$  if  $(\clubsuit)$  admits a solution with  $x_i = x_j$  whenever  $x_i$  and  $x_j$  are in the same class in  $\mathcal{P}$ . We call the linear subspace defined by the latter condition  $\mathbb{R}_{\mathcal{P}}$ .

Having obtained a lifting partition of the ground variational problem, we can now restrict the solution space to  $\{x : Ax \leq b\} \cap \mathbb{R}_{\mathcal{P}}$ . That is, we constrain equivalent variables to be equal, knowing that at least one solution will be preserved in this space of lower dimension. Since ground variables of the same class are now equal, they can be replaced with a single aggregated (lifted) variable. The resulting lifted problem has one variable per equivalence class, thus, if the lifting partition is coarse enough, significant compression and in turn run-time savings can be achieved. To recover a ground solution from the lifted solution, one assigns the value of the lifted variable to every ground variable in the class.

For linear programs, Mladenov et al. [15, 17, 16, 5] have shown that equitable partitions [3] act as lifting partitions. An **equitable partition of a graph** is a partition  $\mathcal{P}$  of the vertex set such that for any pair of vertices  $u$  and  $v$  in the same class  $P_n$  and any other class  $P_m$ ,  $|\text{nb}(u) \cap P_m| = |\text{nb}(v) \cap P_m|$ <sup>1</sup>. For colored graphs, we additionally require that the color vector respects the partition. We call the quantity  $|\text{nb}(v) \cap P_m|$  the degree of  $P_n$  to  $P_m$ ,  $\text{deg}(P_n, P_m)$ . For notational convenience, we will introduce equitable partitions of factor graphs as  $\mathcal{P} = \{P_1, \dots, P_p, Q_1, \dots, Q_q, D_1, \dots, D_d\}$ , where the  $P$ -classes refer to the variable classes, the  $Q$ -classes to factor classes and the  $D$ -classes to position classes.

For the purposes of our discussion, an **equitable partition of a linear program** is an equitable partition of its factor graph. The existence of an equitable partition of a linear program implies the existence of certain doubly-stochastic

<sup>1</sup>The orbit partitions discussed in [1, 2] are a special kind of equitable partitions.

matrices  $(\Sigma, \Pi)$  such that  $\mathbf{c}^T \Pi = \mathbf{c}^T$ ,  $\Sigma \mathbf{b} = \mathbf{b}$  and  $\Sigma \mathbf{A} = \mathbf{A} \Pi$  [5].

Recall that a concave energy inference problem as defined here is essentially a linear program (the MAP-LP) plus a linear combination of local entropies in the objective. The symmetries of the MAP-LP have already received attention [1, 17, 16], and it is understood that the MAP-LP preserves the symmetries present in the model. We will use this understanding as a starting point for our discussion of concave energies. We briefly formalize the claim.

**Lemma 1.** *Any equitable partition  $\mathcal{P}$  of an MRF  $G$  induces an equitable partition  $\mathcal{P}'$  of the resulting MAP-LP.*

Let us briefly sketch how this works. Suppose we are given an equitable partition  $\mathcal{P}$  of  $G$ . To obtain an equitable partition  $\mathcal{P}'$  on  $\mathcal{L}(G)$ , we group together  $\mu_i(0)$  with  $\mu_j(0)$ , resp.  $\mu_i(1)$  with  $\mu_j(1)$  if  $X_i$  is grouped with  $X_j$  in  $\mathcal{P}$ . To partition the joint state pseudomarginals, we use the following rule. Let  $\theta_\alpha$  and  $\theta_\beta$  be two factors grouped together in  $\mathcal{P}$  ( $\alpha = \beta$  is also allowed). Then, for all permutations  $\pi : \alpha \rightarrow \beta$  such that  $\pi(i) = j$  only if  $\diamond_{i\alpha}$  is grouped together with  $\diamond_{j\beta}$  in  $\mathcal{P}$ , we group together  $\mu_\alpha(\mathbf{x})$  with  $\mu_\beta(\pi(\mathbf{x}))$  for every joint configuration  $\mathbf{x}$ . This grouping of the LP variables also induces a grouping of the constraints and positions that completes the partition.

Due to lack of space, we will not prove this here. Instead, we give an example of how model symmetry propagates to MAP-LP symmetry. Fig. 2 shows an MRF  $G$  colored by its CEP (a) and the resulting MAP-LP (b) (some constraints and the position nodes have been omitted for clarity). In Fig. 2c, we see the correspondence between the CEP of  $G$  and the CEP of the MAP-LP as indicated by the colors.

### 3 EQUITABLE PARTITIONS OF CONCAVE FREE ENERGIES

With the basics of lifted variational inference at hand, we can now begin our main discussion. We proceed as follows. We start off by defining an equitable partition of a concave energy. Then, as the first main result of this section, we show that any concave energy that admits an equitable partition has a solution that **respects** that partition. This establishes that equitable partitions of concave energies are lifting partitions. Next, we show that given an equitable partition of an MRF, concave energies that admit this partition are guaranteed to exist. That is, if we want to do convergent inference on a model with symmetries, we can always find a suitable energy that does not break the symmetries. Finally, we will look at some heuristics used for selecting concave energies and examine their relationship to equitable partitions.

**Definition 2.** *An equitable partition of a concave energy  $\hat{H}_c$  (as in Eq. 3) with  $\mathbf{c} \in \mathcal{C}(G)$  is an equitable partition of  $G$  such that  $X_i$  and  $X_j$  are grouped together only if  $c_i = c_j$*

and  $\theta_\alpha, \theta_\beta$  are grouped together only if  $c_\alpha = c_\beta$ .

We will shortly show that equitable partitions of concave energies preserve optimal solutions of the variational problem. Before we do so, however, we will formalize what we consider to be the symmetries of  $\hat{H}_c$ .

If we set aside the constraints and ignore semantics of  $\mu$ ,  $\hat{H}_c$  is just a linear combination of  $x \log x$  terms, i.e.  $\hat{H}_c(\mathbf{x}) = \sum_k c_k x_k \log x_k$ . Observe that if we permute any two variables whose  $c$ 's are the same, we do not change  $\hat{H}_c$ . In other words, any permutation  $\Pi$  with  $\Pi \mathbf{c} = \mathbf{c}$  is an automorphism of  $\hat{H}_c$ , i.e.,  $\hat{H}_c(\Pi \mathbf{x}) = \hat{H}_c(\mathbf{x})$ . Moreover, switching any pair of variables can be done independently of other pairs. We now restate the above in formal terms. If we introduce  $\mathcal{R} = \{R_1, \dots, R_r\}$  that partitions the variables into classes having equal  $c$ , then the following holds:

**Observation 3.** *The group  $\Gamma = \bigotimes_{R \in \mathcal{R}} \mathbb{S}_{|R|}$  is isomorphic to a subgroup of  $\mathbb{AUT}(\hat{H})$ .*

Here  $\bigotimes$  denotes the group product and  $\mathbb{S}_n$  is the symmetric group over  $n$  elements. Now, let us “switch on” the semantics of the argument and interpret the automorphism group of  $\hat{H}_c$  in terms of pseudomarginals. We can see that the following operations are automorphisms of  $\hat{H}_c$ : we can exchange the pseudomarginals of any two variables with the same  $c$ 's, e.g.,  $(\mu_i(0), \mu_i(1)) \mapsto (\mu_j(0), \mu_j(1))$ ; similarly, we can exchange any two sets of factor beliefs; we could also exchange states within a set of beliefs:  $\mu_i(0) \mapsto \mu_i(1)$ , or even exchange states across variables,  $\mu_i(0) \mapsto \mu_j(1)$  (given that the  $c$ 's are compatible). All of these operations can be done independently of each other. Of course, many symmetries of  $\hat{H}_c$  are not symmetries of Eq. 1, as they are not symmetries of the constraints. For example, Eq. 1 would generally not admit, say the reordering of states of a variable without reordering the states of its neighbors as a symmetry, since marginalization constraints tie adjacent nodes in the factor graph.

In summary,  $\hat{H}_c$  is a highly symmetric object given that we have equal  $c$ 's. Hence, what we really have to be careful about are the symmetries of the constraints. However, as we will discuss now, equitability takes care of the constraints and we end up with lifting partitions for Eq. 1.

**Theorem 4** (EPs of Concave Energies are Lifting Partitions). *Let  $G$  be an MRF and  $\mathbf{c} \in \mathcal{C}(G)$  a vector of counting numbers. If  $\mathcal{P}$  is an equitable partition of  $\hat{H}_c$ , then  $\mathcal{P}'$  obtained from  $\mathcal{P}$  via Lemma 1 is a lifting partition of Eq. 1.*

*Proof.* To prove that  $\mathcal{P}'$  is a lifting partition of Eq. 1, we need to prove that Eq. 1 admits a solution, where equivalent variables take on equal values. We will establish this in the following way. Given any feasible vector  $\mu$  of Eq. 1, we will produce a vector  $\mu'$  by replacing each variable by the average of its class. E.g., if the variable  $\mu_i(x)$  is in some class  $P$ , then  $\mu'_i(x) = 1/|P| \sum_{j \in P} \mu_j(x)$ . Thus averaged,

the vector  $\mu'$  respects  $\mathcal{P}'$ . Then, we need that **a)**  $\mu'$  is feasible as well and that **b)**  $F(\mu') \geq F(\mu)$ . Having established **a)** and **b)**, the rest is simple: we take any optimal solution of Eq. 1 and average over the classes. By **a)** we know the average is feasible. By **b)** we know that it will not decrease the objective value. Since we started with something that was already optimal, it must be that averaged vector is of equal objective value, as improvement over the optimum is not possible by definition. Thus we have found a new optimum that respects the partition. It now only remains to verify that **a)** and **b)** indeed hold.

**Proof of a).** The averaging operation over the partition classes can be represented in a linear algebraic way, as multiplying  $\mu$  with the doubly stochastic matrix  $X$  defined as:

$$X_{ij} = \begin{cases} 1/|C| & \text{if } (\mu)_i, (\mu)_j \text{ are both in some } C \in \mathcal{P}', \\ 0 & \text{otherwise.} \end{cases}$$

The brackets in the equation indicate that the indices above are used in a generic sense (not bound to factors, variables or particular states). The theory of equitable partitions of LPs tells us that if  $\mathcal{P}'$  is equitable, then  $\mu$  being feasible implies  $\mu' = X\mu$  is feasible as well, as  $X$  is a fractional automorphism of the LP [5].

**Proof of b).** The Birkhoff-von-Neumann Theorem [12] allows one to decompose the doubly stochastic  $X$  as a convex combination of permutation matrices  $\sum_i \lambda_i \Pi_i$ . Note that any of these permutation matrices will exchange only variables that had been grouped together in  $\mathcal{P}'$ . This follows from the fact that the  $\lambda$ 's form a convex combination. If  $\Pi_k$  has a nonzero element, then  $\sum_k \lambda_k (\Pi_k)_{ij} = X_{ij}$  has to be strictly greater than 0 as well, as the convex combination has at least one nonzero element. By definition  $c$  respects  $\mathcal{P}'$ , hence all  $\Pi$ 's are automorphisms of  $\hat{H}_c$ . Moreover,  $\theta$  also respects  $\mathcal{P}'$  due to our definition of equitable partitions, hence the  $\Pi$ 's are automorphisms of  $\theta^T$  as well. Taken together, we establish that all  $\Pi_i$ 's are automorphisms of  $F$ ,  $F(\mu) = F(\Pi_i \mu)$ . With this in mind, the concavity of  $F$  gives us the result:  $F(\mu') = F(\sum_i \lambda_i \Pi_i \mu) \geq \sum_i \lambda_i F(\Pi_i \mu) = F(\mu)$ .  $\square$

Thus, we have established that equitable partitions of concave energies are lifting partitions for the variational problem of interest. However, an important question that remains is: do they actually exist? That is, if we want to do inference, can we find counting numbers that permit lifting at all? As we will show now, not only do such numbers exist, but also most heuristics presented in literature for finding counting numbers will yield  $c$  that respect equitable partitions of  $G$ . Let us first show the existence of liftable counting numbers.

**Lemma 5** (Existence of liftable counting numbers). *Let  $\mathcal{P}$  be an equitable partition of the MRF  $G$ . If  $\mathcal{C}(G)$  is not empty, then there exists a  $c$ -vector that respects  $\mathcal{P}$ . In other words,  $\mathcal{P}$  is an equitable partition of  $\hat{H}_c$  for at least one  $c$ .*

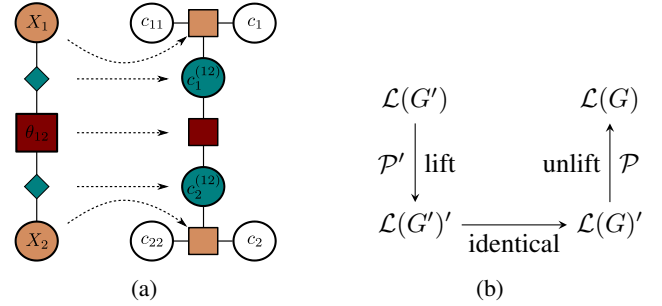


Figure 3: (a) Illustration of Lemma 5. (b) Commutative diagram underlying the “lifted inference by reparametrization” paradigm. (Best viewed in color)

*Proof (sketch).* Observe that the set of counting numbers is defined by a linear program, as Eq. 4 consists of linear constraints. Thus, we can again rely on the fact that equitable partitions of linear programs act as lifting partitions, given that we manage to translate  $\mathcal{P}$  into an equitable partition  $\mathcal{P}'$  of  $\mathcal{C}(G)$ . Let us show the translation in question. We group together  $c_i$  with  $c_j$ ,  $c_{ii}$  with  $c_{jj}$ , and  $\phi_i$  (the constraint generated by  $X_i$ ) with  $\phi_j$  in  $\mathcal{P}'$  if  $X_i$  and  $X_j$  are grouped together in  $\mathcal{P}$ . Similarly, if  $\mathcal{P}$  groups  $\theta_\alpha$  and  $\theta_\beta$ , we group  $c_\alpha$  with  $c_\beta$ ,  $c_{\alpha\alpha}$  with  $c_{\beta\beta}$ , and  $\phi_\alpha$  (the constraint generated by  $\theta_\alpha$ ) with  $\phi_\beta$ . Finally,  $c_{i\alpha}$  and  $c_{j\beta}$  are grouped together if  $\diamond_{i\alpha}$  and  $\diamond_{j\beta}$  are grouped together in  $\mathcal{P}$ . Now let us argue that  $\mathcal{P}'$  is indeed equitable on  $\mathcal{C}(G)$ . The main idea is as follows: we will show that the factor graph of  $G$  is isomorphic to a “skeleton” subgraph of the factor graph of  $\mathcal{C}(G)$ . As such, any equitable partition of  $G$  is also equitable on the skeleton of  $\mathcal{C}(G)$ . Then, we will complete the partition on the remaining elements of  $\mathcal{C}(G)$  in a way that preserves equitability.

To obtain the skeleton, we temporarily ignore the variables  $c_i$ ,  $c_{ii}$ ,  $c_\alpha$ ,  $c_{\alpha\alpha}$ . Then, the map  $M : G \rightarrow \mathcal{C}(G)$ , which maps the position node  $\diamond_{\alpha i}$  to the LP variable  $c_{i\alpha}$ , the factor  $\theta_\alpha$  to the constraint of the factor and the variable  $X_i$  to the constraint of the variable, is an isomorphism. This follows directly from Eq. 4. An LP variable  $c_{i\alpha}$  appears in the factor constraint of  $\theta_\alpha$  if and only if  $\theta_\alpha$  is connected to position node  $\diamond_{\alpha i}$  (in other words variable  $i$  participates in factor  $\theta_\alpha$ ). Moreover a variable  $c_{i\alpha}$  appears in the variable constraint of  $X_i$  if and only if  $X_i$  is connected to position node  $\diamond_{\alpha i}$ . Thus, an equitable partition of  $G$  yields an equitable partition of the constraints and the variables  $c_{i\alpha}$ . If we reintroduce now  $c_i$ ,  $c_{ii}$ ,  $c_\alpha$ ,  $c_{\alpha\alpha}$ , we see that each appears in exactly one constraint, so they can be partitioned in a way to preserve equitability. Fig. 3a provides an illustration. Note that we have so far ignored the position nodes in the FG of  $\mathcal{C}(G)$ . It can be verified that they can be partitioned without breaking the equitability of the partition obtained thus far.  $\square$

We have established that if  $G$  admits an equitable parti-



tion, then there will be at least one energy that admits the same equitable partition. The question now is, how do we compute the appropriate counting numbers? Naturally, as in Thm 4, we can take any vector of counting numbers, any equitable partition of  $G$  and simply average  $c$  over the classes. However, as  $\mathcal{C}(G)$  is a polyhedron, there are infinitely many vectors of counting numbers and the usefulness of the resulting energies in terms of the approximate inference problem will vary. In the following, we show that several heuristics for picking counting numbers naturally yield counting numbers that allow nice equitable partitions of the energy. The heuristics that we will discuss first are the following two: **(a)** [6] following the principle of insufficient reason one tries to make  $c$ 's as uniform as possible by minimizing either  $\sum_{\alpha}(c_{\alpha} - 1)^2$  or  $\sum_{\alpha} c_{\alpha} \log c_{\alpha}$  over  $\mathcal{C}(G)$ ; **(b)** [14] finding the least-squares projection of the  $c$ 's of the Bethe energy onto  $\mathcal{C}(G)$ , i.e.  $c^* = \operatorname{argmin}_{c \in \mathcal{C}(G)} \sum_{\alpha}(c_{\alpha} - 1)^2 + \sum_i (c_i - |\operatorname{nb}(X_i)| + 1)^2$ . Both of them lead to liftable energies.

**Proposition 6** (Finding lifted counting numbers). *Let  $\mathcal{P}$  be an EP of  $G$  and  $\mathcal{P}'$  is an EP of  $\mathcal{C}(G)$  obtained as in Thm. 5. Then  $\mathcal{P}'$  is a lifting partition of both (a) and (b).*

*Proof.* The proof is identical to the proof of Thm. 4. As, Lemma 5 already established, given any  $c \in \mathcal{C}(G)$ , we obtain a new  $c' \in \mathcal{C}(G)$  by assigning to any  $c'_{\alpha}$  or  $c'_i$  the average of  $c$  over the respective class. Now we need to show that this averaging does not increase the respective objective values. We argue in the same way as in Thm. 4. For both cases in **(a)**, the automorphism group consists of the product of two symmetric groups - we can exchange any two  $c_i, c_j$  and any two  $c_{\alpha}, c_{\beta}$  independently. Thus averaging over  $\mathcal{P}'$  is equivalent to averaging over automorphisms of the objectives, which by convexity does not increase the value. For the case of **(b)**, the automorphism group is smaller. We are allowed to exchange any two  $c_{\alpha}, c_{\beta}$  independently, but we can exchange  $c_i, c_j$  only if  $|\operatorname{nb}(X_i)| = |\operatorname{nb}(X_j)|$ . Recall however, that in any equitable partition, for any class  $Q$ , we group  $X_i$  and  $X_j$  if  $\deg(X_i, Q) = \deg(X_j, Q)$ . This implies  $|\operatorname{nb}(X_i)| = |\operatorname{nb}(X_j)|$ . Thus, the averaging matrix  $X$  as in Thm. 4 will have a non-zero value only if  $|\operatorname{nb}(X_i)| = |\operatorname{nb}(X_j)|$ , and the resulting Birkhoff-von-Neumann decomposition will consist of automorphisms of the objective.  $\square$

In essence, this proposition tells us that optimal numbers (w.r.t. **(a)** and **(b)**) can be found that turn any equitable partition of  $G$  into an equitable partition of the energy. Could this be the case for all heuristics? As it turns out, some heuristics can impose restrictions on what partitions can be EPs of their respective energies.

To see this, consider for example as a further option **(c)** Tree-Reweighted BP (in pairwise models). We obtain  $c \in \mathcal{C}(G)$  by setting  $c_{\alpha}$  to be the number of spanning trees passing through  $\theta_{\alpha}$  divided by the number of all spanning trees in  $G$ . If we translate the result of Bui et al. [2] in the lan-

guage of the present paper, it states that the equitable partition of  $G$  coarser than its orbit partition may generally not be turned into an equitable partition of the TRW energy. However, for the orbit partition, they give an efficient algorithm to produce the appropriate  $c$ .

Finally, one could also follow Meshi et al. [14] as option **(d)**: instead of approximating the numbers of  $F_{\text{Bethe}}$ , we project  $F_{\text{Bethe}}$  itself onto the set of concave energies  $F_{c \in \mathcal{C}(G)}$ . More precisely, we take  $c^* = \operatorname{argmin}_{c \in \mathcal{C}(G)} \int_{\mu \in \mathcal{L}(G)} (F_{\text{Bethe}}(\mu) - F_c(\mu))^2 d\mu$ . As a matter of fact, it is even an open question whether  $F_{\text{Bethe}}$  admits stationary points that respect<sup>2</sup> any  $\mathcal{P}$ . It should be noted that if we want a coarser partition than what **(c)** or **(d)** permit, we could still take the counting numbers and average them over a coarser equitable partition of  $G$ . However, the question of whether this operation preserves the quality of approximation remains open.

## 4 LIFTING AS REPARAMETRIZATION

One issue that arises with the above approach is that in certain cases, compression changes the structure of the optimization problem. For example, given an equitable partition  $\mathcal{P} = \{Q_1, \dots, Q_q, P_1, \dots, P_p, D_1, \dots, D_d\}$  of  $G$ , the compressed inference problem will take on the following form:  $\mu^* =$

$$\operatorname{argmax}_{\mu \in \mathcal{L}(G)'} E'(\mu) + \sum_{P \in \mathcal{P}} |P| c_P H_P(\mu_P) + \sum_{Q \in \mathcal{P}} |Q| c_Q H_Q(\mu_Q),$$

where

$$E'(\mu) = \sum_{P \in \mathcal{P}} |P| \theta_P \mu_P + \sum_{Q \in \mathcal{P}} |Q| \theta_Q \mu_Q \quad \text{and}$$

$$\mathcal{L}(G)' = \left\{ \mu \geq 0 \left| \begin{array}{l} \mu_P^0 + \mu_{P'}^1 = 1 \\ \mu_Q^{00} + \mu_D^{01} = \mu_P^0 \\ \mu_Q^{11} + \mu_{D'}^{01} = \mu_{P'}^1 \\ \mu_Q^{00} + \mu_{D'}^{01} = \mu_P^0 \\ \mu_Q^{11} + \mu_D^{01} = \mu_{P'}^1 \end{array} \right. \right\}. \quad (5)$$

Here,  $Q$  is the representative of a factor class,  $P$  and  $P'$  are the representatives of its neighboring variable classes and  $D$  and  $D'$  are the representatives of the position classes that connect variables in  $P$  and  $P'$  to factors in  $Q$ . Note that our notation treats the beliefs of factor  $Q$  being in state 00 or 11 differently from the beliefs of factor  $Q$  being in state 01 or 01. This becomes important in the following situation: in an equitable partition, it could happen that  $P = P'$ . That is, for any ground factor in the class  $Q$ , both participating variables are in the same class of the lifting partition. For the variational problem, this means that we need to unify the

<sup>2</sup>Works on lifted loopy belief propagation [8, 22, 10] do indeed show that BP admits such solutions. However, the question of whether this is due to  $F_{\text{Bethe}}$  or an artifact of the way BP optimizes it is unclear.

variables  $\mu_{D'}^{01}$  and  $\mu_D^{01}$ , ending up with a peculiar-looking set of constraints such as:

$$\mu_Q^{00} + \mu_D^{01} = \mu_P^0 \text{ and } \mu_Q^{11} + \mu_D^{01} = \mu_P^1 \quad .$$

This example illustrates why we cannot simply view the lifted inference problem as a standard inference problem on a smaller “proper” factor graph. We have a binary factor that has the same variable in both positions (which is, unfortunately, not equivalent to a unary factor). Now, in terms of message-passing, to send a message to  $P$ ,  $Q$  would have to first eliminate  $P$  – an operation which is not supported by standard message-passing frameworks.

To circumvent the problem, at least for the case of  $T = 0$  in Eq. 1, we can follow the “lifted inference as reparametrization” paradigm recently advocated by Mladenov et al. [16]. The main idea is based on the fact that equitable partitions allow one to not only recover a solution of the ground variational problem from the lifted one, but also a solution of the ground problem can be projected onto  $\mathbb{R}_{\mathcal{P}}$  by averaging the variables within each equivalence class. Thus, what we do is the following: given an  $G$  and an equitable partition  $\mathcal{P}$ , we find a smaller  $G'$  and a partition  $\mathcal{P}'$  such that the variational problem of  $G$  lifted with  $\mathcal{P}$  is identical to the variational problem of  $G'$  lifted with  $\mathcal{P}'$ . So, we solve the smaller  $G'$  with a ground solver, average over  $\mathcal{P}'$  to obtain a solution to the lifted problem of  $G'$ , transfer that to the lifted variational problem of  $G$ , since they are identical, and then finally unlift according to  $\mathcal{P}$ . The idea is illustrated in Fig. 3. The work of [16] shows how to find  $G'$  and  $\mathcal{P}$  and then reparametrize the factors of  $G'$  such that its lifted MAP-LPs are same the one of  $G$ . Our goal here is to show that if in addition we reparametrize the  $c$ 's, their lifted concave energies will also be the same. Thus, in essence, we can make use of any ground concave energy solver that allows manually setting  $c$ 's for lifted inference.

As we would like to avoid introducing again the technical arguments of [16], we will introduce a notion of weak partition equivalence, which subsumes the equivalence in [16], and build upon that. We will show that reparametrization of  $\hat{H}_c$  is possible among partition-equivalent pairs. Since the pairs produced by the algorithm of [16] are partition-equivalent, the result applies.

**Definition 7.** Let  $G$  and  $G'$  be MRFs. We call  $G$  and  $G'$  **weakly partition-equivalent** if they admit equitable partitions  $\mathcal{P} = \{P_1, \dots, P_p, Q_1, \dots, Q_q, D_1, \dots, D_d\}$  resp.  $\mathcal{P}' = \{P'_1, \dots, P'_p, Q'_1, \dots, Q'_q, D'_1, \dots, D'_d\}$  having the same number of variable, factor and position classes. Moreover, for any two classes  $X, Y \in \mathcal{P}$ , we have  $\deg(X, Y) \neq 0$  if and only if  $\deg(X', Y') \neq 0$  ( $\spadesuit$ ).

Note that the actual number of nodes in the class may very well be different, we do not require  $|Q| = |Q'|$ .

**Lemma 8.** If  $G$  and  $G'$  are weakly partition equivalent w.r.t.  $\mathcal{P}$  and  $\mathcal{P}'$ , then  $\deg(Q, D) \neq 0$  implies

$$\frac{|Q|}{|Q'|} \deg(Q, D) = \frac{|D|}{|D'|} \deg(Q', D'), \text{ resp. } \deg(P, D) \neq 0 \text{ implies } \frac{|P|}{|P'|} \deg(P, D) = \frac{|D|}{|D'|} \deg(P', D').$$

*Proof.* For any equitable partition it will hold that  $|Q| \deg(Q, D) = |D| \deg(D, Q)$ , resp.  $|Q'| \deg(Q', D') = |D'| \deg(D, Q)$ . Since  $D$  consists of only position nodes, and every position node can be connected to exactly one factor (and one variable),  $\deg(D, Q) = \deg(D', Q') = 1$ . With this in mind, we take the quotient of both equations and obtain  $\frac{|Q| \deg(Q, D)}{|Q'| \deg(Q', D')} = \frac{|D|}{|D'|}$ . Multiplying both sides by  $\deg(Q', D')$  yields the result. The reasoning for the variable classes is identical.  $\square$

Now, suppose counting numbers for  $G$  which respect  $\mathcal{P}$  have been found, that is, for every  $\theta_\alpha$  in the class  $Q$ ,  $c_\alpha = c_Q$  and so on, as in Thm. 5. Then, we can construct a vector of counting numbers for  $G'$  by the following procedure: for every vertex in  $G'$  (regardless of whether it is a variable, factor, or position), we take the size of its class,  $|X'|$ , the size of the corresponding class in  $\mathcal{P}$ ,  $|X|$  and then normalize the counting number  $c_X$  (from  $G$ ) by their ratio. That is,  $c'_k = (|X|/|X'|)c_X$ .

**Theorem 9.** Suppose  $G$  and  $G'$  are weakly partition equivalent with respect to  $\mathcal{P}$  and  $\mathcal{P}'$  and  $c \in \mathcal{C}(G)$  respects  $\mathcal{P}$ . Then, the vector  $c'$  having  $c'_k = c_{X'} = \frac{|X|}{|X'|}c_X$  consists of counting numbers for  $G'$ . I.e.  $c' \in \mathcal{C}(G')$ .

*Proof.* We have assumed  $c$  respects  $\mathcal{P}$ . This allows us to rewrite the conditions of Eq. 4 as

$$C_Q = C_{QQ} + \sum_D \deg(Q, D)c_D \quad \text{and,} \\ C_P = C_{PP} + \sum_D \deg(P, D)c_D .$$

The above describes the lifted LP of  $\mathcal{C}(G)$  after unification of all equivalent variables. Now, observe that

$$C_{Q'} = \frac{|Q|}{|Q'|} C_Q = \frac{|Q|}{|Q'|} \left[ C_{QQ} + \sum_D \deg(Q, D)c_D \right] \\ = \frac{|Q|}{|Q'|} C_{QQ} + \sum_{D: \deg(Q, D) \neq 0} \frac{|Q|}{|Q'|} \deg(Q, D)c_D \\ = C_{Q'Q'} + \sum_{D: \deg(Q, D) \neq 0} \deg(Q', D') \frac{|D|}{|D'|} c_D \\ = C_{Q'Q'} + \sum_{D': \deg(Q', D') \neq 0} \deg(Q', D') c_{D'} .$$

Note, we were allowed to switch the index in the last line due to ( $\spadesuit$ ). Similarly,

$$C_{P'} = \frac{|P|}{|P'|} C_P = \frac{|P|}{|P'|} \left[ C_{PP} - \sum_D \deg(P, D)c_D \right] \\ = \frac{|P|}{|P'|} C_{PP} - \sum_{D: \deg(P, D) \neq 0} \frac{|P|}{|P'|} \deg(P, D)c_D \\ = C_{P'P'} - \sum_{D: \deg(P, D) \neq 0} \deg(P', D') \frac{|D|}{|D'|} c_D \\ = C_{P'P'} - \sum_{D': \deg(P', D') \neq 0} \deg(P', D') c_{D'} .$$

|  |   |  |
|--|---|--|
|  | $W \quad x \neq y \wedge \neg Fr(x, y)$                   |  |
|  | -0.8 $Ca(x)$  | $W \quad x \neq y \wedge (Q1(x) \Leftrightarrow \neg Q2(y))$ |
|  | -12.4 $Aux(x, y)$   | $W \quad x \neq y \wedge (Q2(x) \Leftrightarrow \neg Q3(y))$ |
| $W \quad V(x)$                                     | 1.5 $(Sm(x) \Rightarrow Ca(x))$                           | $W \quad x \neq y \wedge (Q3(x) \Leftrightarrow \neg Q1(y))$ |
| -0.1 $x \neq y \wedge (V(x) \Leftrightarrow V(y))$ | 6.2 $(x \neq y \wedge Aux(x, y) \wedge Smokes(x))$        | -W $x \neq y \wedge (Q1(x) \Leftrightarrow Q2(y))$           |
|  | 6.2 $(x \neq y \wedge Aux(x, y) \wedge Smokes(y))$        | -W $x \neq y \wedge (Q2(x) \Leftrightarrow Q3(y))$           |
|  | 6.2 $(x \neq y \wedge Aux(x, y) \wedge Friends(x, y))$    | -W $x \neq y \wedge (Q3(x) \Leftrightarrow Q1(y))$           |
|  | -3.1 $(x \neq y \wedge (Smokes(x) \wedge Friends(x, y)))$ |  |
|  | -3.1 $(x \neq y \wedge (Smokes(y) \wedge Friends(x, y)))$ |  |
| Complete Graph                                     | Friends-Smokers   | Clique-Cycle   |

Table 1: The Markov Logic Network models used to illustrate our theoretical results. Details are given in the main text.

What this tells us is that  $c'$  is a solution of  $\mathcal{C}(G')$  lifted according to  $\mathcal{P}'$ . This implies that we can recover a vector of ground counting numbers by assigning to  $c'_k$  the corresponding  $c'_X$ .  $\square$

So now, given  $c$  obtained as above, let us examine the lifted entropy of  $G'$  with respect to  $\mathcal{P}'$ ,

$$\begin{aligned} \hat{H}' &= \sum_{P' \in \mathcal{P}'} |P'| c_{P'} H_{P'} + \sum_{Q' \in \mathcal{P}'} |Q'| c_{Q'} H_{Q'} \\ &= \sum_{P' \in \mathcal{P}'} |P'| \frac{|P|}{|P'|} c_P H_{P'} + \sum_{Q' \in \mathcal{P}'} |Q'| \frac{|Q|}{|Q'|} c_Q H_{Q'} \\ &= \sum_{P \in \mathcal{P}} |P| c_P H_P + \sum_{Q \in \mathcal{P}} |Q| c_Q H_Q = \hat{H}. \end{aligned}$$

As we see, this is exactly the lifted entropy of  $G$  with respect to  $\mathcal{P}$ . To conclude the argument, we note that the algorithm of [16] takes care that the linear  $E'(\cdot)$  part and the constraints of the lifted variational problems are the same. Thus, with the addition of the reparametrized  $c$ 's, we achieve full equivalence of both lifted variational problems.

## 5 EMPIRICAL ILLUSTRATION

We will now illustrate our theoretical results by demonstrating that the “lifting by reparametrization” paradigm allows one to lift Schwing et al.’s distributed message-passing algorithm<sup>3</sup> for marginal inference [21] without any overhead. As far as we know, this presents the first lifted convergent variational marginal inference that can handle problems efficiently by distributing and parallelizing the inference computation and the memory requirements. The convergence and optimality guarantees are preserved by consistency messages, sent between the distributed cores, that our lifted variant directly inherits from Schwing et al.’s original version. Together with parallelizable lifting approaches [11], this shows for the first time that each step of the lifted inference pipeline is readily parallelizable.

The experimental protocol is inspired by [2] and uses three of their test models in Markov Logic Network syntax, cf. Tab. 1: complete graph, friends-smokers, and clique-cycle.

We focus on the repulsive case, i.e. the weight of interaction clauses is set to a negative values. The parameter  $W$  denotes the weight that will be varying across the experiments. As Bui et al. [2] point out, in all models except clique-cycle,  $W$  acts like the “local field” potential in an Ising model; a positive value of  $W$  means that variables tend to be in the 1 state, whereas a negative value favors the 0 state. The complete graph model is an Ising model over the complete graph over  $n$  nodes (the domain size); all parameters are the same. The weight of the interaction clause is repulsive with  $-0.1$ . The friends-smokers is a pairwise version of the one negated one used by Bui et al., where we also used a repulsive interaction between smokers: Here the domain size is the number of people. Finally, the clique-cycle model encodes a model with 3 cliques, each consisting of  $n$  nodes (the domain size), and 3 bipartite graphs between them. For more details we refer to [2].

Since we reparametrize an existing variational approach, we will not report on the accuracy of the marginals and the quality of the objective as they have been investigated already in the corresponding literature. Rather, we perform a “within model” comparison of the objectives achieved, the size of the models, and the running times for inference since they are what our theoretical results are about. Specifically, we evaluated the lifted and the ground version on several instances of the three test models, varying the parameter  $W$  and the domain size. We assume no evidence has been observed, which results in a large amount of symmetries. As Bui et al. [2] argue this is a sensible setting since performing marginal inference in relational probabilistic models can be very useful for maximum-likelihood parameter estimation.

The experimental results are summarized in Fig. 4. As one can see the lifted inference can be orders of magnitude faster than its ground version. We also ran a single-core lifted loopy belief propagation. However, due to its lack of convergence, it is difficult to have a meaningful comparison of running times. Nevertheless, we observed cases where it did not converge but actually started to oscillate.

<sup>3</sup><http://alexander-schwing.de/projects.php>

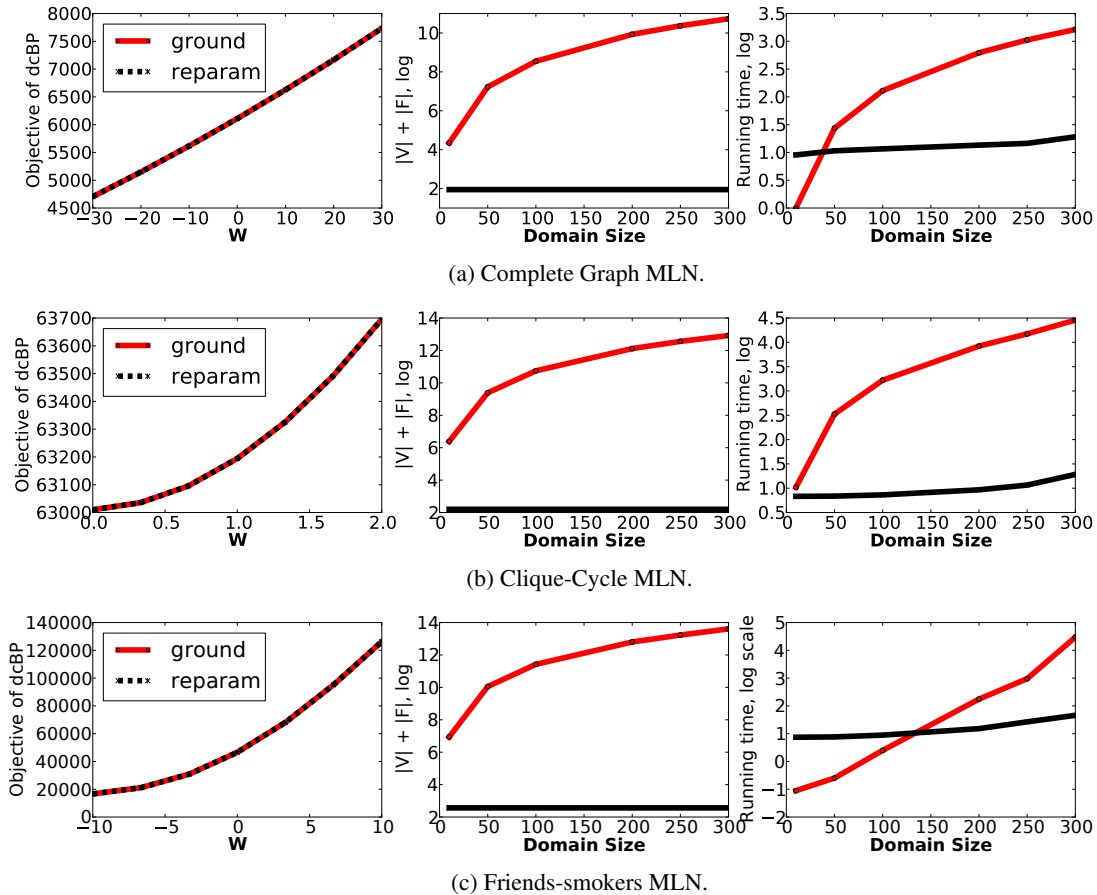


Figure 4: Experimental results on the test models from Tab. 1. Each row shows from left to right the objective for different weights, the size (number of nodes and factors) in log-space and the running time in seconds in log-space for ground (red) versus lifted (black). As one case see, lifted variational marginal inference can be orders of magnitude faster than it ground version without sacrificing the objective. (best viewed in color)

## 6 CONCLUSIONS

We have established a “lifted inference by reparametrization” paradigm for variational marginal inference. More precisely, we have introduced the notion of equitable partitions of concave free energies and shown how to use them to reparameterize the corresponding variational optimization problems. In turn, a large class of existing variational marginal inference algorithms can directly be made aware of symmetries without modifications. We illustrated this by lifting Schwing et al.’s distributed message-passing algorithm for marginal inference, resulting in the first lifted, distributed, convergent message passing algorithms for marginal inference. Moreover, the paradigm of reparametrization allows us to address the observation of Bui et al. [2] about their Frank-Wolfe TRW solver running slower than BP. At least in the case where no extra tightening is required, one can just compute the TRW counting numbers with lifted Kruskal, reparametrize and apply a generic convergent message-passing algorithm.

Our work provides several avenues for future work. For instance, one should explore what other constraints we can pose on counting numbers to enforce exactness while we can still optimize over the set in a lifted fashion. Since the dimensionality reduction changes the geometry of the variational optimization problem, one should also investigate its interaction with the solvers. It is interesting to explore features of relational languages to speed up lifted variational marginal inference even more. One of the most interesting open question raised by our work is whether non-trivial reparametrizations of  $F_{\text{Bethe}}$  and of energies in general exists and are exploitable for speeding up optimization, at least in an approximate sense. An affirmative answer would have deep implications not only for probabilistic inference but for many tasks in computer vision, machine learning, and AI in general.

**Acknowledgements:** The authors would like to thank the reviewers for their feedback. This research was partly supported by the German-Israeli Foundation (GIF) for Scientific Research and Development, 1180-218.6/2011.

## References

- [1] H.H. Bui, T.N. Huynh, and S. Riedel. Automorphism groups of graphical models and lifted variational inference. In *Proc. of the 29th Conference on Uncertainty in Artificial Intelligence (UAI-2013)*, 2013.
- [2] H.H. Bui, T.N. Huynh, and D. Sontag. Lifted tree-reweighted variational inference. In *Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2014.
- [3] C. Godsil and G. Royle. *Algebraic Graph Theory*. Springer, 2001.
- [4] Eric Gribkoff, Guy Van den Broeck, and Dan Suciu. Understanding the complexity of lifted inference and asymmetric weighted model counting. In *Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence (UAI)*, July 2014.
- [5] M. Grohe, K. Kersting, M. Mladenov, and E. Selman. Dimension reduction via colour refinement. In *Proceedings of the 22th Annual European Symposium on Algorithms (ESA)*, pages 505–516, 2014.
- [6] T. Hazan and A. Shashua. Convergent message-passing algorithms for inference over general graphs with convex free energies. In *Proceedings of the Twenty-Forth Conference in Uncertainty in Artificial Intelligence, (UAI)*, pages 264–273, 2008.
- [7] T. Heskes. Convexity arguments for efficient minimization of the bethe and kikuchi free energies. *J. Artif. Intell. Res. (JAIR)*, 26:153–190, 2006.
- [8] A. Jaimovich, O. Meshi, and N. Friedman. Template-based Inference in Symmetric Relational Markov Random Fields. In *Proc. of the Conf. on Uncertainty in Artificial Intelligence (UAI-07)*, pages 191–199, 2007.
- [9] K. Kersting. Lifted probabilistic inference. In *Proceedings of ECAI-2012*. IOS Press, 2012.
- [10] K. Kersting, B. Ahmadi, and S. Natarajan. Counting Belief Propagation. In *Proc. of the 25th Conf. on Uncertainty in Artificial Intelligence (UAI-09)*, 2009.
- [11] K. Kersting, M. Mladenov, R. Garnett, and M. Grohe. Power iterated color refinement. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI)*, pages 1904–1910, 2014.
- [12] Albert W. Marshall, Ingram Olkin, and Barry C. Arnold. *Inequalities : theory of majorization and its applications*. Springer series in statistics. Springer, New York, 2011.
- [13] T. Meltzer, A. Globerson, and Y. Weiss. Convergent message passing algorithms - a unifying view. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 393–401, 2009.
- [14] O. Meshi, A. Jaimovich, A. Globerson, and N. Friedman. Convexifying the bethe free energy. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 402–410, 2009.
- [15] M. Mladenov, B. Ahmadi, and K. Kersting. Lifted linear programming. In *15th Int. Conf. on Artificial Intelligence and Statistics (AISTATS 2012)*, pages 788–797, 2012. Volume 22 of JMLR: W&CP 22.
- [16] M. Mladenov, A. Globerson, and K. Kersting. Lifted message passing as reparametrization of graphical models. In *Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2014.
- [17] M. Mladenov, K. Kersting, and A. Globerson. Efficient lifting of MAP LP relaxations using k-locality. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics (AISTATS) 22-25, 2014*, pages 623–632, 2014.
- [18] M. Niepert. Markov chains on orbits of permutation groups. In *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2012.
- [19] M. Niepert and G. Van den Broeck. Tractability through exchangeability: A new perspective on efficient probabilistic inference. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI)*, pages 2467–2475, 2014.
- [20] D. Poole. First-Order Probabilistic Inference. In *Proc. of the 18th International Joint Conference on Artificial Intelligence (IJCAI-05)*, pages 985–991, 2003.
- [21] A.G. Schwing, T. Hazan, M. Pollefeys, and R. Urtasun. Distributed message passing for large scale graphical models. In *Proceedings of the 24th IEEE Conference on Computer Vision and Pattern Recognition, (CVPR)*, pages 1833–1840, 2011.
- [22] P. Singla and P. Domingos. Lifted First-Order Belief Propagation. In *Proc. of the 23rd AAAI Conf. on Artificial Intelligence (AAAI-08)*, pages 1094–1099, Chicago, IL, USA, July 13-17 2008.
- [23] G. Van den Broeck and M. Niepert. Lifted probabilistic inference for asymmetric graphical models. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI)*, 2015.
- [24] M.J. Wainwright and M.I. Jordan. Graphical models, exponential families, and variational inference. *Found. Trends Mach. Learn.*, 1(1-2):1–305, January 2008.

---

# Non-parametric Revenue Optimization for Generalized Second Price Auctions

---

**Mehryar Mohri**

Courant Institute and Google Research,  
251 Mercer Street, New York, NY

**Andrés Muñoz Medina**

Courant Institute of Mathematical Sciences,  
251 Mercer Street, New York, NY

## Abstract

We present an extensive analysis of the key problem of learning optimal reserve prices for generalized second price auctions. We describe two algorithms for this task: one based on density estimation, and a novel algorithm benefiting from solid theoretical guarantees and with a very favorable running-time complexity of  $O(nS \log(nS))$ , where  $n$  is the sample size and  $S$  the number of slots. Our theoretical guarantees are more favorable than those previously presented in the literature. Additionally, we show that even if bidders do not play at an equilibrium, our second algorithm is still well defined and minimizes a quantity of interest. To our knowledge, this is the first attempt to apply learning algorithms to the problem of reserve price optimization in GSP auctions. Finally, we present the first convergence analysis of empirical equilibrium bidding functions to the unique symmetric Bayesian-Nash equilibrium of a GSP.

## 1 INTRODUCTION

The Generalized Second-Price (GSP) auction is currently the standard mechanism used for selling sponsored search advertisement. As suggested by the name, this mechanism generalizes the standard second-price auction of [Vickrey \(1961\)](#) to multiple items. In the case of sponsored search advertisement, these items correspond to ad slots which have been ranked by their position. Given this ranking, the GSP auction works as follows: first, each advertiser places a bid; next, the seller, based on the bids placed, assigns a score to each bidder. The highest scored advertiser is assigned to the slot in the best position, that is, the one with the highest likelihood of being clicked on. The second highest score obtains the second best item and so on, until all slots have been allocated or all advertisers have been assigned to a slot. As with second-price auctions, the bidder's payment is independent of his bid. Instead, it depends

solely on the bid of the advertiser assigned to the position below.

In spite of its similarity with second-price auctions, the GSP auction is not an incentive-compatible mechanism, that is, bidders have an incentive to lie about their valuations. This is in stark contrast with second-price auctions where truth revealing is in fact a dominant strategy. It is for this reason that predicting the behavior of bidders in a GSP auction is challenging. This is further worsened by the fact that these auctions are repeated multiple times a day. The study of all possible equilibria of this repeated game is at the very least difficult. While incentive compatible generalizations of the second-price auction exist, namely the Vickrey-Clark-Gloves (VCG) mechanism, the simplicity of the payment rule for GSP auctions as well as the large revenue generated by them has made the adoption of VCG mechanisms unlikely.

Since its introduction by Google, GSP auctions have generated billions of dollars across different online advertisement companies. It is therefore not surprising that it has become a topic of great interest for diverse fields such as Economics, Algorithmic Game Theory and more recently Machine Learning.

The first analysis of GSP auctions was carried out independently by [Edelman et al. \(2005\)](#) and [Varian \(2007\)](#). Both publications considered a full information scenario, that is one where the advertisers' valuations are publicly known. This assumption is weakly supported by the fact that repeated interactions allow advertisers to infer their adversaries' valuations. [Varian \(2007\)](#) studied the so-called Symmetric Nash Equilibria (SNE) which is a subset of the Nash equilibria with several favorable properties. In particular, Varian showed that any SNE induces an efficient allocation, that is an allocation where the highest positions are assigned to advertisers with high values. Furthermore, the revenue earned by the seller when advertisers play an SNE is always at least as much as the one obtained by VCG. The authors also presented some empirical results showing that some bidders indeed play by using an SNE. However, no theoretical justification can be given for the choice of

this subset of equilibria (Börger et al., 2013; Edelman and Schwarz, 2010). A finer analysis of the full information scenario was given by Lucier et al. (2012). The authors proved that, excluding the payment of the highest bidder, the revenue achieved at any Nash equilibrium is at least one half that of the VCG auction.

Since the assumption of full information can be unrealistic, a more modern line of research has instead considered a Bayesian scenario for this auction. In a Bayesian setting, it is assumed that advertisers’ valuations are i.i.d. samples drawn from a common distribution. Gomes and Sweeney (2014) characterized all symmetric Bayes-Nash equilibria and showed that any symmetric equilibrium must be efficient. This work was later extended by Sun et al. (2014) to account for the quality score of each advertiser. The main contribution of this work was the design of an algorithm for the crucial problem of revenue optimization for the GSP auction. Lahaie and Pennock (2007) studied different *squashing* ranking rules for advertisers commonly used in practice and showed that none of these rules are necessarily optimal in equilibrium. This work is complemented by the simulation analysis of Vorobeychik (2009) who quantified the distance from equilibrium of bidding truthfully. Lucier et al. (2012) showed that the GSP auction with an optimal reserve price achieves at least 1/6 of the optimal revenue (of any auction) in a Bayesian equilibrium. More recently, Thompson and Leyton-Brown (2013) compared different allocation rules and showed that an *anchoring* allocation rule is optimal when valuations are sampled i.i.d. from a uniform distribution. With the exception of Sun et al. (2014), none of these authors have proposed an algorithm for revenue optimization using historical data.

Zhu et al. (2009) introduced a ranking algorithm to learn an optimal allocation rule. The proposed ranking is a convex combination of a quality score based on the features of the advertisement as well as a revenue score which depends on the value of the bids. This work was later extended in (He et al., 2014) where, in addition to the ranking function, a behavioral model of the advertisers is learned by the authors.

The rest of this paper is organized as follows. In Section 2, we give a learning formulation of the problem of selecting reserve prices in a GSP auction. In Section 3, we discuss previous work related to this problem. Next, we present and analyze two learning algorithms for this problem in Section 4, one based on density estimation extending to this setting an algorithm of Guerre et al. (2000), and a novel discriminative algorithm taking into account the loss function and benefiting from favorable learning guarantees. Section 5 provides a convergence analysis of the empirical equilibrium bidding function to the true equilibrium bidding function in a GSP. On its own, this result is of great interest as it justifies the common assumption of buyers playing a symmetric Bayes-Nash equilibrium. Finally, in

Section 6, we report the results of experiments comparing our algorithms and demonstrating in particular the benefits of the second algorithm.

## 2 MODEL

For the most part, we will use the model defined by Sun et al. (2014) for GSP auctions with incomplete information. We consider  $N$  bidders competing for  $S$  slots with  $N \geq S$ . Let  $v_i \in [0, 1]$  and  $b_i \in [0, 1]$  denote the per-click valuation of bidder  $i$  and his bid respectively. Let the position factor  $c_s \in [0, 1]$  represent the probability of a user noticing an ad in position  $s$  and let  $e_i \in [0, 1]$  denote the expected click-through rate of advertiser  $i$ . That is  $e_i$  is the probability of ad  $i$  being clicked on given that it was noticed by the user. We will adopt the common assumption that  $c_s > c_{s+1}$  (Gomes and Sweeney, 2014; Lahaie and Pennock, 2007; Sun et al., 2014; Thompson and Leyton-Brown, 2013). Define the score of bidder  $i$  to be  $s_i = e_i v_i$ . Following Sun et al. (2014), we assume that  $s_i$  is an i.i.d. realization of a random variable with distribution  $F$  and density function  $f$ . Finally, we assume that advertisers bid in an efficient symmetric Bayes-Nash equilibrium. This is motivated by the fact that even though advertisers may not infer what the valuation of their adversaries is from repeated interactions, they can certainly estimate the distribution  $F$ .

Define  $\pi: s \mapsto \pi(s)$  as the function mapping slots to advertisers, i.e.  $\pi(s) = i$  if advertiser  $i$  is allocated to position  $s$ . For a vector  $\mathbf{x} = (x_1, \dots, x_N) \in \mathbb{R}^N$ , we use the notation  $x^{(s)} := x_{\pi(s)}$ . Finally, denote by  $r_i$  the reserve price for advertiser  $i$ . An advertiser may participate in the auction only if  $b_i \geq r_i$ . In this paper we present an analysis of the two most common ranking rules (Qin et al., 2014):

1. Rank-by-bid. Advertisers who bid above their reserve price are ranked in descending order of their bids and the payment of advertiser  $\pi(s)$  is equal to  $\max(r^{(s)}, b^{(s+1)})$ .
2. Rank-by-revenue. Each advertiser is assigned a quality score  $q_i := q_i(b_i) = e_i b_i \mathbb{1}_{b_i \geq r_i}$  and the ranking is done by sorting these scores in descending order. The payment of advertiser  $\pi(s)$  is given by  $\max(r^{(s)}, \frac{q^{(s+1)}}{e^{(s)}})$ .

In both setups, only advertisers bidding above their reserve price are considered. Notice that rank-by-bid is a particular case of rank-by-revenue where all quality scores are equal to 1. Given a vector of reserve prices  $\mathbf{r}$  and a bid vector  $\mathbf{b}$ , we define the revenue function to be

$$\begin{aligned} \text{Rev}(\mathbf{r}, \mathbf{b}) &= \sum_{s=1}^S c_s \left( \frac{q^{(s+1)}}{e^{(s)}} \mathbb{1}_{q^{(s+1)} \geq e^{(s)} r^{(s)}} + r^{(s)} \mathbb{1}_{q^{(s+1)} < e^{(s)} r^{(s)}} \right) \end{aligned}$$

Using the notation of [Mohri and Medina \(2014\)](#), we define the loss function

$$L(\mathbf{r}, \mathbf{b}) = -\text{Rev}(\mathbf{r}, \mathbf{b}).$$

Given an i.i.d. sample  $\mathcal{S} = (\mathbf{b}_1, \dots, \mathbf{b}_n)$  of realizations of an auction, our objective will be to find a reserve price vector  $\mathbf{r}^*$  that maximizes the expected revenue. Equivalently,  $\mathbf{r}^*$  should be a solution of the following optimization problem:

$$\min_{\mathbf{r} \in [0,1]^N} \mathbb{E}_{\mathbf{b}}[L(\mathbf{r}, \mathbf{b})]. \quad (1)$$

### 3 PREVIOUS WORK

It has been shown, both theoretically and empirically, that reserve prices can increase the revenue of an auction ([Myerson, 1981](#); [Ostrovsky and Schwarz, 2011](#)). The choice of an appropriate reserve price therefore becomes crucial. If it is chosen too low, the seller might lose some revenue. On the other hand, if it is set too high, then the advertisers may not wish to bid above that value and the seller will not obtain any revenue from the auction.

[Mohri and Medina \(2014\)](#), [Pardoe et al. \(2005\)](#), and [Cesa-Bianchi et al. \(2013\)](#) have given learning algorithms that estimate the optimal reserve price for a second-price auction in different information scenarios. The scenario we consider is most closely related to that of [Mohri and Medina \(2014\)](#). An extension of this work to the GSP auction, however, is not straightforward. Indeed, as we will show later, the optimal reserve price vector depends on the distribution of the advertisers' valuation. In a second-price auction, these valuations are observed since the corresponding mechanism is an incentive-compatible. This does not hold for GSP auctions. Moreover, for second-price auctions, only one reserve price had to be estimated. In contrast, our model requires the estimation of up to  $N$  parameters with intricate dependencies between them.

The problem of estimating valuations from observed bids in a non-incentive compatible mechanism has been previously analyzed. [Guerre et al. \(2000\)](#) described a way of estimating valuations from observed bids in a first-price auction. We will show that this method can be extended to the GSP auction. The rate of convergence of this algorithm, however, in general will be worse than the standard learning rate of  $O(\frac{1}{\sqrt{n}})$ .

[Sun et al. \(2014\)](#) showed that, for advertisers playing an efficient equilibrium, the optimal reserve price is given by  $r_i = \frac{\bar{r}}{e_i}$  where  $\bar{r}$  satisfies

$$\bar{r} = \frac{1 - F(\bar{r})}{f(\bar{r})}.$$

The authors suggest learning  $\bar{r}$  via a maximum likelihood technique over some parametric family to estimate  $f$  and

$F$ , and to use these estimates in the above expression. There are two main drawbacks for this algorithm. The first is a standard problem of parametric statistics: there are no guarantees on the convergence of their estimation procedure when the density function  $f$  is not part of the parametric family considered. While this problem can be addressed by the use of a non-parametric estimation algorithm such as kernel density estimation, the fact remains that the function  $f$  is the density for the unobservable scores  $s_i$  and therefore cannot be properly estimated. The solution proposed by the authors assumes that the bids in fact form a perfect SNE and so advertisers' valuations can be recovered using the process described by [Varian \(2007\)](#). There is however no justification for this assumption and, in fact, we show in [Section 6](#) that bids played in a Bayes-Nash equilibrium do not in general form a SNE.

### 4 LEARNING ALGORITHMS

Here, we present and analyze two algorithms for learning the optimal reserve price for a GSP auction when advertisers play a symmetric equilibrium.

#### 4.1 DENSITY ESTIMATION ALGORITHM

First, we derive an extension of the algorithm of [Guerre et al. \(2000\)](#) to GSP auctions. To do so, we first derive a formula for the bidding strategy at equilibrium. Let  $z_s(v)$  denote the probability of winning position  $s$  given that the advertiser's valuation is  $v$ . It is not hard to verify that

$$z_s(v) = \binom{N-1}{s-1} (1 - F(v))^{s-1} F^p(v),$$

where  $p = N - s$ . Indeed, in an efficient equilibrium, the bidder with the  $s$ -th highest valuation must be assigned to the  $s$ -th highest position. Therefore an advertiser with valuation  $v$  is assigned to position  $s$  if and only if  $s-1$  bidders have a higher valuation and  $p$  have a lower valuation.

For a rank-by-bid auction, [Gomes and Sweeney \(2014\)](#) showed the following results.

**Theorem 1** ([Gomes and Sweeney \(2014\)](#)). *A GSP auction has a unique efficient symmetric Bayes-Nash equilibrium with bidding strategy  $\beta$  if and only if  $\beta$  is strictly increasing and satisfies the following integral equation:*

$$\begin{aligned} & \sum_{s=1}^S c_s \int_0^v \frac{dz_s(t)}{dt} t dt \\ &= \sum_{s=1}^S c_s \binom{N-1}{s-1} (1 - F(v))^{s-1} \int_0^v \beta(t) p F^{p-1}(t) f(t) dt. \end{aligned} \quad (2)$$

Furthermore, the optimal reserve price  $r^*$  satisfies

$$r^* = \frac{1 - F(r^*)}{f(r^*)}. \quad (3)$$



The authors show that, if the click probabilities  $c_s$  are sufficiently diverse, then,  $\beta$  is guaranteed to be strictly increasing. When ranking is done by revenue, Sun et al. (2014) gave the following theorem.

**Theorem 2** (Sun et al. (2014)). *Let  $\beta$  be defined by the previous theorem. If advertisers bid in a Bayes-Nash equilibrium then  $b_i = \frac{\beta(v_i)}{e_i}$ . Moreover, the optimal reserve price vector  $\mathbf{r}^*$  is given by  $r_i^* = \frac{\bar{r}}{e_i}$  where  $\bar{r}$  satisfies equation (3).*

We are now able to present the foundation of our first algorithm. Instead of assuming that the bids constitute an SNE as in (Sun et al., 2014), we follow the ideas of Guerre et al. (2000) and infer the scores  $s_i$  only from observables  $b_i$ . Our result is presented for the rank-by-bid GSP auction but an extension to the rank-by-revenue mechanism is trivial.

**Lemma 1.** *Let  $v_1, \dots, v_n$  be an i.i.d. sample of valuations from distribution  $F$  and let  $b_i = \beta(v_i)$  be the bid played at equilibrium. Then the random variables  $b_i$  are i.i.d. with distribution  $G(b) = F(\beta^{-1}(b))$  and density  $g(b) = \frac{f(\beta^{-1}(b))}{\beta'(\beta^{-1}(b))}$ . Furthermore,*

$$\begin{aligned} v_i &= \beta^{-1}(b_i) \\ &= \frac{\sum_{s=1}^S c_s \binom{N-1}{s-1} (1-G(b_i))^{s-1} b_i p G(b_i)^{p-1} g(b_i)}{\sum_{s=1}^S c_s \binom{N-1}{s-1} \frac{d\bar{z}}{db}(b_i)} \\ &\quad - \frac{\sum_{s=1}^S c_s (s-1) (1-G(b_i))^{s-2} g(b_i) \int_0^{b_i} p G(u)^{p-1} u g(u) du}{\sum_{s=1}^S c_s \binom{N-1}{s-1} \frac{d\bar{z}}{db}(b_i)}, \end{aligned} \quad (4)$$

where  $\bar{z}_s(b) := z_s(\beta^{-1}(b))$  and is given by  $\binom{N-1}{s-1} (1-G(b))^{s-1} G(b)^{p-1}$ .

*Proof.* By definition,  $b_i = \beta(v_i)$  is a function of only  $v_i$ . Since  $\beta$  does not depend on the other samples either, it follows that  $(b_i)_{i=1}^N$  must be an i.i.d. sample. Using the fact that  $\beta$  is a strictly increasing function we also have  $G(b) = P(b_i \leq b) = P(v_i \leq \beta^{-1}(b)) = F(\beta^{-1}(b))$  and a simple application of the chain rule gives us the expression for the density  $g(b)$ . To prove the second statement observe that by the change of variable  $v = \beta^{-1}(b)$ , the right-hand side of (2) is equal to

$$\begin{aligned} &\sum_{s=1}^S \binom{N-1}{s-1} (1-G(b))^{s-1} \int_0^{\beta^{-1}(b)} p \beta(t) F^{p-1}(t) f(t) dt \\ &= \sum_{s=1}^S \binom{N-1}{s-1} (1-G(b))^{s-1} \int_0^b p u G(u)^{p-1} g(u) du. \end{aligned}$$

The last equality follows by the change of variable  $t = \beta(u)$  and from the fact that  $g(b) = \frac{f(\beta^{-1}(b))}{\beta'(\beta^{-1}(b))}$ . The same change of variables applied to the left-hand side of (2)

yields the following integral equation:

$$\begin{aligned} &\sum_{s=1}^S \binom{N-1}{s-1} \int_0^b \beta^{-1}(u) \frac{d\bar{z}}{du}(u) du \\ &= \sum_{s=1}^S \binom{N-1}{s-1} (1-G(b))^{s-1} \int_0^b u p G(u)^{p-1} g(u) du. \end{aligned}$$

Taking the derivative with respect to  $b$  of both sides of this equation and rearranging terms lead to the desired expression.  $\square$

The previous Lemma shows that we can recover the valuation of an advertiser from its bid. We therefore propose the following algorithm for estimating the value of  $\bar{r}$ .

1. Use the sample  $\mathcal{S}$  to estimate  $G$  and  $g$ .
2. Plug this estimates in (4) to obtain approximate samples from the distribution  $F$ .
3. Use the approximate samples to find estimates  $\hat{f}$  and  $\hat{F}$  of the valuations density and cumulative distribution functions respectively.
4. Use  $\hat{F}$  and  $\hat{f}$  to estimate  $\bar{r}$ .

In order to avoid the use of parametric methods, a kernel density estimation algorithm can be used to estimate  $g$  and  $f$ . While this algorithm addresses both drawbacks of the algorithm proposed by Sun et al. (2014), it can be shown (Guerre et al., 2000)[Theorem 2] that if  $f$  is  $R$  times continuously differentiable, then, after seeing  $n$  samples,  $\|f - \hat{f}\|_\infty$  is in  $\Omega\left(\frac{1}{n^{R/(2R+3)}}\right)$  independently of the algorithm used to estimate  $f$ . In particular, note that for  $R = 1$  the rate is in  $\Omega\left(\frac{1}{n^{1/4}}\right)$ . This unfavorable rate of convergence can be attributed to the fact that a two-step estimation algorithm is being used (estimation of  $g$  and  $f$ ). But, even with access to bidder valuations, the rate can only be improved to  $\Omega\left(\frac{1}{n^{R/(2R+1)}}\right)$  (Guerre et al., 2000). Furthermore, a small error in the estimation of  $f$  affects the denominator of the equation defining  $\bar{r}$  and can result in a large error on the estimate of  $\bar{r}$ .

## 4.2 DISCRIMINATIVE ALGORITHM

In view of the problems associated with density estimation, we propose to use empirical risk minimization to find an approximation to the optimal reserve price. In particular, we are interested in solving the following optimization problem:

$$\min_{\mathbf{r} \in [0,1]^N} \sum_{i=1}^n L(\mathbf{r}, \mathbf{b}_i). \quad (5)$$

We first show that, when bidders play in equilibrium, the optimization problem (1) can be considerably simplified.

**Proposition 1.** *If advertisers play a symmetric Bayes-Nash equilibrium then*

$$\min_{\mathbf{r} \in [0,1]^N} \mathbb{E}_{\mathbf{b}}[L(\mathbf{r}, \mathbf{b})] = \min_{r \in [0,1]} \mathbb{E}_{\mathbf{b}}[\tilde{L}(r, \mathbf{b})],$$

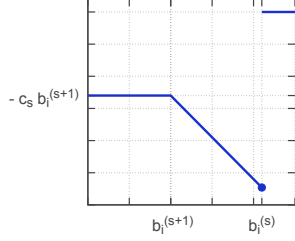


Figure 1: Plot of the loss function  $L_{i,s}$ . Notice that the loss in fact resembles a broken “V”.

where  $\tilde{q}_i := \tilde{q}_i(b_i) = e_i b_i$  and

$$\tilde{L}(r, \mathbf{b}) = - \sum_{s=1}^S \frac{c_s}{e^{(s)}} \left( \tilde{q}^{(s+1)} \mathbb{1}_{\tilde{q}^{(s+1)} \geq r} + r \mathbb{1}_{\tilde{q}^{(s+1)} < r \leq \tilde{q}^{(s)}} \right).$$

*Proof.* Since advertisers play a symmetric Bayes-Nash equilibrium, the optimal reserve price vector  $\mathbf{r}^*$  is of the form  $r_i^* = \frac{r}{e_i}$ . Therefore, letting  $D = \{\mathbf{r} | r_i = \frac{r}{e_i}, r \in [0, 1]\}$  we have  $\min_{\mathbf{r} \in [0, 1]^N} \mathbb{E}_{\mathbf{b}}[L(\mathbf{r}, \mathbf{b})] = \min_{\mathbf{r} \in D} \mathbb{E}_{\mathbf{b}}[L(\mathbf{r}, \mathbf{b})]$ . Furthermore, when restricted to  $D$ , the objective function  $L$  is given by

$$- \sum_{s=1}^S \frac{c_s}{e^{(s)}} \left( q^{(s+1)} \mathbb{1}_{q^{(s+1)} \geq r} + r \mathbb{1}_{q^{(s+1)} < r \leq q^{(s)}} \right).$$

Thus, we are left with showing that replacing  $q^{(s)}$  with  $\tilde{q}^{(s)}$  in this expression does not affect its value. Let  $r \geq 0$ , since  $q_i = \tilde{q}_i \mathbb{1}_{\tilde{q}_i \geq r}$ , in general the equality  $q^{(s)} = \tilde{q}^{(s)}$  does not hold. Nevertheless, if  $s_0$  denotes the largest index less than or equal to  $S$  satisfying  $q^{(s_0)} > 0$ , then  $\tilde{q}^{(s)} \geq r$  for all  $s \leq s_0$  and  $q^{(s)} = \tilde{q}^{(s)}$ . On the other hand, for  $S \geq s > s_0$ ,  $\mathbb{1}_{q^{(s)} \geq r} = \mathbb{1}_{\tilde{q}^{(s)} \geq r} = 0$ . Thus,

$$\begin{aligned} & \sum_{s=1}^S \frac{c_s}{e^{(s)}} \left( q^{(s+1)} \mathbb{1}_{q^{(s+1)} \geq r} + r \mathbb{1}_{q^{(s+1)} < r \leq q^{(s)}} \right) \\ &= \sum_{s=1}^{s_0} \frac{c_s}{e^{(s)}} \left( q^{(s+1)} \mathbb{1}_{q^{(s+1)} \geq r} + r \mathbb{1}_{q^{(s+1)} < r \leq q^{(s)}} \right) \\ &= \sum_{s=1}^{s_0} \frac{c_s}{e^{(s)}} \left( \tilde{q}^{(s+1)} \mathbb{1}_{\tilde{q}^{(s+1)} \geq r} + r \mathbb{1}_{\tilde{q}^{(s+1)} < r \leq \tilde{q}^{(s)}} \right) \\ &= -\tilde{L}(r, \mathbf{b}), \end{aligned}$$

which completes the proof.  $\square$

In view of this proposition, we can replace the challenging problem of solving an optimization problem in  $\mathbb{R}^N$  with solving the following simpler empirical risk minimization problem

$$\min_{r \in [0, 1]} \sum_{i=1}^n \tilde{L}(r, \mathbf{b}_i) = \min_{r \in [0, 1]} \sum_{i=1}^n \sum_{s=1}^S L_{s,i}(r, \tilde{q}^{(s)}, \tilde{q}^{(s+1)}), \quad (6)$$

---

### Algorithm 1 Minimization algorithm

---

**Require:** Scores  $(\tilde{q}_i^{(s)})$ ,  $1 \leq n, 1 \leq s \leq S$ .

- 1: Define  $(p_{is}^{(1)}, p_{is}^{(2)}) = (\tilde{q}_i^{(s)}, \tilde{q}_i^{(s+1)})$ ;  $m = nS$ ;
  - 2:  $\mathcal{N} := \bigcup_{i=1}^n \bigcup_{s=1}^S \{p_{is}^{(1)}, p_{is}^{(2)}\}$ ;
  - 3:  $(n_1, \dots, n_{2m}) = \mathbf{Sort}(\mathcal{N})$ ;
  - 4: Set  $\mathbf{d}_i := (d_1, d_2) = \mathbf{0}$
  - 5: Set  $d_1 = - \sum_{i=1}^n \sum_{s=1}^S \frac{c_s}{e_i} p_{is}^{(2)}$ ;
  - 6: Set  $r^* = -1$  and  $L^* = \infty$
  - 7: **for**  $j = 2, \dots, 2m$  **do**
  - 8:   **if**  $n_{j-1} = p_{is}^{(2)}$  **then**
  - 9:      $d_1 = d_1 + \frac{c_s}{e_i} p_{is}^{(2)}$ ;  $d_2 = d_2 - \frac{c_s}{e_i}$ ;
  - 10:   **else if**  $n_{j-1} = p_{is}^{(1)}$  **then**
  - 11:      $d_2 = d_2 + \frac{c_s}{e_s}$
  - 12:   **end if**
  - 13:    $L = d_1 - n_j d_2$ ;
  - 14:   **if**  $L < L^*$  **then**
  - 15:      $L^* = L$ ;  $r^* = n_j$ ;
  - 16:   **end if**
  - 17: **end for**
  - 18: **return**  $r^*$ ;
- 

where  $L_{s,i}(r, \tilde{q}^{(s)}, \tilde{q}^{(s+1)}) := - \frac{c_s}{e^{(s)}} (\tilde{q}_i^{(s+1)} \mathbb{1}_{\tilde{q}_i^{(s+1)} \geq r} - r \mathbb{1}_{\tilde{q}_i^{(s+1)} < r \leq \tilde{q}_i^{(s)}})$ . In order to efficiently minimize this highly non-convex function, we draw upon the work of [Mohri and Medina \(2014\)](#) on minimization of sums of  $v$ -functions.

**Definition 1.** A function  $V: \mathbb{R}^3 \rightarrow \mathbb{R}$  is a  $v$ -function if it admits the following form:

$$\begin{aligned} & V(r, q_1, q_2) \\ &= -a^{(1)} \mathbb{1}_{r \leq q_2} - a^{(2)} r \mathbb{1}_{q_2 < r \leq q_1} + \left[ \frac{r}{\eta} - a^{(3)} \right] \mathbb{1}_{q_1 < r < (1+\eta)q_1}, \end{aligned}$$

with  $0 \leq a^{(1)}, a^{(2)}, a^{(3)}, \eta \leq \infty$  constants satisfying  $a^{(1)} = a^{(2)} q_2$ ,  $-a^{(2)} q_1 \mathbb{1}_{\eta > 0} = \left( \frac{1}{\eta} q_1 - a^{(3)} \right) \mathbb{1}_{\eta > 0}$ . Under the convention that  $0 \cdot \infty = 0$ .

As suggested by their name, these functions admit a characteristic “V shape”. It is clear from Figure 1 that  $L_{s,i}$  is a  $v$ -function with  $a^{(1)} = \frac{c_s}{e^{(s)}} \tilde{q}_i^{(s+1)}$ ,  $a^{(2)} = \frac{c_s}{e^{(s)}}$  and  $\eta = 0$ . Thus, we can apply the optimization algorithm given by [Mohri and Medina \(2014\)](#) to minimize (6) in  $O(nS \log nS)$  time. Algorithm 1 gives the pseudocode of that the adaptation of this general algorithm to our problem. A proof of the correctness of this algorithm can be found in [\(Mohri and Medina, 2014\)](#).

We conclude this section by presenting learning guarantees for our algorithm. Our bounds are given in terms of the Rademacher complexity and the VC-dimension.

**Definition 2.** Let  $\mathcal{X}$  be a set and let  $G := \{g: \mathcal{X} \rightarrow \mathbb{R}\}$  be a family of functions. Given a sample  $\mathcal{S} = (x_1, \dots, x_n) \in$

$\mathcal{X}$ , the empirical Rademacher complexity of  $G$  is defined by

$$\widehat{\mathfrak{R}}_S(G) = \frac{1}{n} \mathbb{E}_\sigma \left[ \sup_{g \in G} \frac{1}{n} \sum_{i=1}^n \sigma_i g(x_i) \right],$$

where  $\sigma_i$ s are independent random variables distributed uniformly over the set  $\{-1, 1\}$ .

**Proposition 2.** Let  $m = \min_i e_i > 0$  and  $\mathfrak{M} = \sum_{s=1}^S c_s$ . Then, for any  $\delta > 0$ , with probability at least  $1 - \delta$  over the draw of a sample  $\mathcal{S}$  of size  $n$ , each of the following inequalities holds for all  $r \in [0, 1]$ :

$$\mathbb{E}[\widetilde{L}(r, \mathbf{b})] \leq \frac{1}{n} \sum_{i=1}^n \widetilde{L}(r, \mathbf{b}_i) + C(\mathfrak{M}, m, n, \delta) \quad (7)$$

$$\frac{1}{n} \sum_{i=1}^n \widetilde{L}(r, \mathbf{b}_i) \leq \mathbb{E}[\widetilde{L}(r, \mathbf{b})] + C(\mathfrak{M}, m, n, \delta), \quad (8)$$

where  $C(\mathfrak{M}, m, n, \delta) = \frac{1}{\sqrt{n}} + \sqrt{\frac{\log(en)}{n}} + \sqrt{\frac{\mathfrak{M} \log(1/\delta)}{2mn}}$ .

*Proof.* Let  $\Psi: S \mapsto \sup_{r \in [0,1]} \frac{1}{n} \sum_{i=1}^n \widetilde{L}(r, \mathbf{b}_i) - \mathbb{E}[\widetilde{L}(r, \mathbf{b})]$ . Let  $\mathcal{S}^i$  be a sample obtained from  $\mathcal{S}$  by replacing  $\mathbf{b}_i$  with  $\mathbf{b}'_i$ . It is not hard to verify that  $|\Psi(\mathcal{S}) - \Psi(\mathcal{S}^i)| \leq \frac{\mathfrak{M}}{nm}$ . Thus, it follows from a standard learning bound that, with probability at least  $1 - \delta$ ,

$$\mathbb{E}[\widetilde{L}(r, \mathbf{b})] \leq \frac{1}{n} \sum_{i=1}^n \widetilde{L}(r, \mathbf{b}_i) + \widehat{\mathfrak{R}}_S(\mathcal{R}) + \sqrt{\frac{\mathfrak{M} \log(1/\delta)}{2mn}},$$

where  $\mathcal{R} = \{\bar{L}_r : \mathbf{b} \mapsto \widetilde{L}(r, \mathbf{b}) | r \in [0, 1]\}$ . We proceed to bound the empirical Rademacher complexity of the class  $\mathcal{R}$ . For  $q_1 > q_2 \geq 0$  let  $\bar{L}(r, q_1, q_2) = q_2 \mathbb{1}_{q_2 > r} + r \mathbb{1}_{q_1 \geq r \geq q_2}$ . By definition of the Rademacher complexity we can write

$$\begin{aligned} \widehat{\mathfrak{R}}_S(\mathcal{R}) &= \frac{1}{n} \mathbb{E}_\sigma \left[ \sup_{r \in [0,1]} \sum_{i=1}^n \sigma_i \bar{L}_r(\mathbf{b}_i) \right] \\ &= \frac{1}{n} \mathbb{E}_\sigma \left[ \sup_{r \in [0,1]} \sum_{i=1}^n \sigma_i \sum_{s=1}^S \frac{c_s}{e_s} \bar{L}(r, \tilde{q}_i^{(s)}, \tilde{q}_i^{(s+1)}) \right] \\ &\leq \frac{1}{n} \mathbb{E}_\sigma \left[ \sum_{s=1}^S \sup_{r \in [0,1]} \sum_{i=1}^n \sigma_i \psi_s(\bar{L}(r, \tilde{q}_i^{(s)}, \tilde{q}_i^{(s+1)})) \right], \end{aligned}$$

where  $\psi_s$  is the  $\frac{c_s}{m}$ -Lipschitz function mapping  $x \mapsto \frac{c_s}{e^{(s)}} x$ . Therefore, by Talagrand's contraction lemma (Ledoux and Talagrand, 2011), the last term is bounded by

$$\sum_{s=1}^S \frac{c_s}{nm} \mathbb{E}_\sigma \sup_{r \in [0,1]} \sum_{i=1}^n \sigma_i \bar{L}(r, \tilde{q}_i^{(s)}, \tilde{q}_i^{(s+1)}) = \sum_{s=1}^S \frac{c_s}{m} \widehat{\mathfrak{R}}_{\mathcal{S}_s}(\widetilde{\mathcal{R}}),$$

where  $\mathcal{S}_s = ((\tilde{q}_1^{(s)}, \tilde{q}_1^{(s+1)}), \dots, (\tilde{q}_n^{(s)}, \tilde{q}_n^{(s+1)}))$  and  $\widetilde{\mathcal{R}} := \{\bar{L}(r, \cdot, \cdot) | r \in [0, 1]\}$ . The loss  $\bar{L}(r, \tilde{q}^{(s)}, \tilde{q}^{(s+1)})$  in fact evaluates to the negative revenue of a second-price auction

with highest bid  $\tilde{q}^{(s)}$  and second highest bid  $\tilde{q}^{(s+1)}$  (Mohri and Medina, 2014). Therefore, by Propositions 9 and 10 of Mohri and Medina (2014) we can write

$$\begin{aligned} \widehat{\mathfrak{R}}_{\mathcal{S}_s}(\widetilde{\mathcal{R}}) &\leq \frac{1}{n} \mathbb{E}_\sigma \left[ \sup_{r \in [0,1]} \sum_{i=1}^n r \sigma_i \right] + \sqrt{\frac{2 \log en}{n}} \\ &\leq \left( \frac{1}{\sqrt{n}} + \sqrt{\frac{2 \log en}{n}} \right), \end{aligned}$$

which concludes the proof.  $\square$

**Corollary 1.** Under the hypotheses of Proposition 2, let  $\hat{r}$  denote the empirical minimizer and  $r^*$  the minimizer of the expected loss. Then, for any  $\delta > 0$ , with probability at least  $1 - \delta$ , the following inequality holds:

$$\mathbb{E}[\widetilde{L}(\hat{r}, \mathbf{b})] - \mathbb{E}[\widetilde{L}(r^*, \mathbf{b})] \leq 2C\left(\mathfrak{M}, m, n, \frac{\delta}{2}\right).$$

*Proof.* By the union bound, (7) and (8) hold simultaneously with probability at least  $1 - \delta$  if  $\delta$  is replaced by  $\delta/2$  in those expression. Adding both inequalities and using the fact that  $\hat{r}$  is an empirical minimizer yields the result.  $\square$

It is worth noting that our algorithm is well defined whether or not the buyers bid in equilibrium. Indeed, the algorithm consists of the minimization over  $r$  of an observable quantity. While we can guarantee convergence to a solution of (1) only when buyers play a symmetric BNE, our algorithm will still find an approximate solution to

$$\min_{r \in [0,1]} \mathbb{E}_{\mathbf{b}}[L(r, \mathbf{b})],$$

which remains a quantity of interest that can be close to (1) if buyers are close to the equilibrium.

## 5 CONVERGENCE OF EMPIRICAL EQUILIBRIA

A crucial assumption in the study of GSP auctions, including this work, is that advertisers bid in a Bayes-Nash equilibrium (Lucier et al., 2012; Sun et al., 2014). This assumption is partially justified by the fact that advertisers can infer the underlying distribution  $F$  using as observations the outcomes of the past repeated auctions and can thereby implement an efficient equilibrium.

In this section, we provide a stronger theoretical justification in support of this assumption: we quantify the difference between the bidding function calculated using observed empirical distributions and the true symmetric bidding function in equilibria. For the sake of notation simplicity, we will consider only the rank-by-bid GSP auction.

Let  $\mathcal{S}_v = (v_1, \dots, v_n)$  be an i.i.d. sample of values drawn from a continuous distribution  $F$  with density function  $f$ .

Assume without loss of generality that  $v_1 \leq \dots \leq v_n$  and let  $\mathbf{v}$  denote the vector defined by  $\mathbf{v}_i = v_i$ . Let  $\widehat{F}$  denote the empirical distribution function induced by  $\mathcal{S}_v$  and let  $\mathbf{F} \in \mathbb{R}^n$  and  $\mathbf{G} \in \mathbb{R}^n$  be defined by  $\mathbf{F}_i = \widehat{F}(v_i) = i/n$  and  $\mathbf{G}_i = 1 - \mathbf{F}_i$ .

We consider a *discrete* GSP auction where the advertiser's valuations are i.i.d. samples drawn from a distribution  $\widehat{F}$ . In the event where two or more advertisers admit the same valuation, ties are broken randomly. Denote by  $\widehat{\beta}$  the bidding function for this auction in equilibrium (when it exists). We are interested in characterizing  $\widehat{\beta}$  and in providing guarantees on the convergence of  $\widehat{\beta}$  to  $\beta$  as the sample size increases.

We first introduce the notation used throughout this section.

**Definition 3.** Given a vector  $\mathbf{F} \in \mathbb{R}^n$ , the backwards difference operator  $\Delta : \mathbb{R}^n \rightarrow \mathbb{R}^n$  is defined as:

$$\Delta \mathbf{F}_i = \mathbf{F}_i - \mathbf{F}_{i-1},$$

for  $i > 1$  and  $\Delta \mathbf{F}_1 = \mathbf{F}_1$ .

We will denote  $\Delta \Delta \mathbf{F}_i$  by  $\Delta^2 \mathbf{F}_i$ . Given any  $k \in \mathbb{N}$  and a vector  $\mathbf{F}$ , the vector  $\mathbf{F}^k$  is defined as  $\mathbf{F}_i^k = (\mathbf{F}_i)^k$ . Let us now define the discrete analog of the function  $z_s$  that quantifies the probability of winning slot  $s$ .

**Proposition 3.** In a symmetric efficient equilibrium of the discrete GSP, the probability  $\widehat{z}_s(v)$  that an advertiser with valuation  $v$  is assigned to slot  $s$  is given by

$$\begin{aligned} \widehat{z}_s(v) &= \sum_{j=0}^{N-s} \sum_{k=0}^{s-1} \binom{N-1}{j, k, N-1-j-k} \frac{\mathbf{F}_{i-1}^j \mathbf{G}_i^k}{(N-j-k)n^{N-1-j-k}}, \end{aligned}$$

if  $v = v_i$  and otherwise by

$$\widehat{z}_s(v) = \binom{N-1}{s-1} \lim_{v' \rightarrow v^-} \widehat{F}(v')^p (1 - \widehat{F}(v))^{s-1} =: \widehat{z}_s^-(v),$$

where  $p = N - s$ .

In particular, notice that  $\widehat{z}_s^-(v_i)$  admits the simple expression

$$\widehat{z}_s^-(v_i) = \binom{N-1}{s-1} \mathbf{F}_{i-1}^p \mathbf{G}_{i-1}^{s-1},$$

which is the discrete version of the function  $z_s$ . On the other hand, even though  $\widehat{z}_s(v_i)$  does not admit a closed-form, it is not hard to show that

$$\widehat{z}_s(v_i) = \binom{N-1}{s-1} \mathbf{F}_{i-1}^p \mathbf{G}_i^{s-1} + O\left(\frac{1}{n}\right). \quad (9)$$

Which again can be thought of as a discrete version of  $z_s$ . The proof of this and all other propositions in this section

are deferred to the Appendix. Let us now define the lower triangular matrix  $\mathbf{M}(s)$  by:

$$\mathbf{M}_{ij}(s) = -\binom{N-1}{s-1} \frac{n \Delta \mathbf{F}_j^p \Delta \mathbf{G}_i^s}{s},$$

for  $i > j$  and

$$\mathbf{M}_{ii}(s) = \sum_{j=0}^{N-s-1} \sum_{k=0}^{s-1} \binom{N-1}{j, k, N-1-j-k} \frac{\mathbf{F}_{i-1}^j \mathbf{G}_i^k}{(N-j-k)n^{N-1-j-k}}.$$

**Proposition 4.** If the discrete GSP auction admits a symmetric efficient equilibrium, then its bidding function  $\widehat{\beta}$  satisfies  $\widehat{\beta}(v_i) = \beta_i$ , where  $\beta$  is the solution of the following linear equation.

$$\mathbf{M}\beta = \mathbf{u}, \quad (10)$$

with  $\mathbf{M} = \sum_{s=1}^S c_s \mathbf{M}(s)$  and  $\mathbf{u}_i = \sum_{s=1}^S \left( c_s z_s(v_i) v_i - \sum_{j=1}^i \widehat{z}_s^-(v_j) \Delta \mathbf{v}_j \right)$ .

To gain some insight about the relationship between  $\widehat{\beta}$  and  $\beta$ , we compare equations (10) and (2). An integration by parts of the right-hand side of (2) and the change of variable  $G(v) = 1 - F(v)$  show that  $\beta$  satisfies

$$\begin{aligned} \sum_{s=1}^S c_s v z_s(v) - \int_0^v \frac{dz_s(t)}{dt} t dt &= \sum_{s=1}^S c_s \binom{N-1}{s-1} G(v)^{s-1} \int_0^v \beta(t) dF^p. \end{aligned} \quad (11)$$

On the other hand, equation (10) implies that for all  $i$

$$\mathbf{u}_i = \sum_{s=1}^S c_s \left[ \mathbf{M}_{ii}(s) \beta_i - \binom{N-1}{s-1} \frac{n \Delta \mathbf{G}_i^s}{s} \sum_{j=1}^{i-1} \Delta \mathbf{F}_j^p \beta_j \right]. \quad (12)$$

Moreover, by Lemma 2 and Proposition 10 in the Appendix, the equalities  $-\frac{n \Delta \mathbf{G}_i^s}{s} = \mathbf{G}_i^{s-1} + O\left(\frac{1}{n}\right)$  and

$$\mathbf{M}_{ii}(s) = \frac{1}{2n} \binom{N-1}{s-1} p \mathbf{F}_{i-1}^{p-1} \mathbf{G}_i^{s-1} + O\left(\frac{1}{n^2}\right),$$

hold. Thus, equation (12) resembles a numerical scheme for solving (11) where the integral on the right-hand side is approximated by the trapezoidal rule. Equation (11) is in fact a Volterra equation of the first kind with kernel

$$K(t, v) = \sum_{s=1}^S \binom{N-1}{s-1} G(v)^{s-1} p F^{p-1}(t).$$

Therefore, we could benefit from the extensive literature on the convergence analysis of numerical schemes for this type of equations (Baker, 1977; Kress et al., 1989; Linz, 1985). However, equations of the first kind are in general

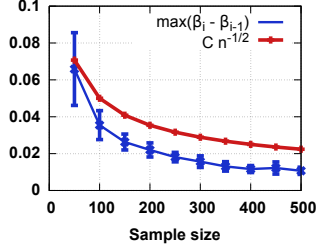


Figure 2: (a) Empirical verification of Assumption 2. The blue line corresponds to the quantity  $\max_i \Delta\beta_i$ . In red we plot the desired upper bound for  $C = 1/2$ .

ill-posed problems (Kress et al., 1989), that is small perturbations on the equation can produce large errors on the solution. When the kernel  $K$  satisfies  $\min_{t \in [0,1]} K(t, t) > 0$ , there exists a standard technique to transform an equation of the first kind to an equation of the second kind, which is a well posed problem. Thus, making the convergence analysis for these types of problems much simpler. The kernel function appearing in (11) does not satisfy this property and therefore these results are not applicable to our scenario. To the best of our knowledge, there exists no quadrature method for solving Volterra equations of the first kind with vanishing kernel.

In addition to dealing with an uncommon integral equation, we need to address the problem that the elements of (10) are not exact evaluations of the functions defining (11) but rather stochastic approximations of these functions. Finally, the grid points used for the numerical approximation are also random.

In order to prove convergence of the function  $\hat{\beta}$  to  $\beta$  we will make the following assumptions

**Assumption 1.** *There exists a constant  $c > 0$  such that  $f(x) > c$  for all  $x \in [0, 1]$ .*

This assumption is needed to ensure that the difference between consecutive samples  $v_i - v_{i-1}$  goes to 0 as  $n \rightarrow \infty$ , which is a necessary condition for the convergence of any numerical scheme.

**Assumption 2.** *The solution  $\beta$  of (10) satisfies  $v_i, \beta_i \geq 0$  for all  $i$  and  $\max_{i \in 1, \dots, n} \Delta\beta_i \leq \frac{C}{\sqrt{n}}$ , for some universal constant  $C$ .*

Since  $\beta_i$  is a bidding strategy in equilibrium, it is reasonable to expect that  $v_i \geq \beta_i \geq 0$ . On the other hand, the assumption on  $\Delta\beta_i$  is related to the smoothness of the solution. If the function  $\beta$  is smooth, we should expect the approximation  $\hat{\beta}$  to be smooth too. Both assumptions can in practice be verified empirically, Figure 2 depicts the quantity  $\max_{i \in 1, \dots, n} \Delta\beta_i$  as a function of the sample size  $n$ .

**Assumption 3.** *The solution  $\beta$  to (2) is twice continuously differentiable.*

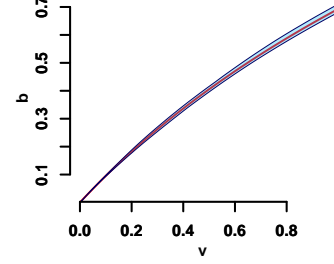


Figure 3: Approximation of the empirical bidding function  $\hat{\beta}$  to the true solution  $\beta$ . The true solution is shown in red and the shaded region represents the confidence interval of  $\hat{\beta}$  when simulating the discrete GSP 10 times with a sample of size 200. Where  $N = 3$ ,  $S = 2$ ,  $c_1 = 1$ ,  $c_2 = 0.5$  and bids were sampled uniformly from  $[0, 1]$

This is satisfied if for instance the distribution function  $F$  is twice continuously differentiable. We can now present our main result.

**Theorem 3.** *If Assumptions 1, 2 and 3 are satisfied, then, for any  $\delta > 0$ , with probability at least  $1 - \delta$  over the draw of a sample of size  $n$ , the following bound holds for all  $i \in [1, n]$ :*

$$|\hat{\beta}(v_i) - \beta(v_i)| \leq e^C \left[ \frac{\log(\frac{2}{\delta})^{\frac{N}{2}}}{\sqrt{n}} q\left(n, \frac{2}{\delta}\right)^3 + \frac{Cq(n, \frac{2}{\delta})}{n^{3/2}} \right].$$

where  $q(n, \delta) = \frac{2}{c} \log(nc/2\delta)$  with  $c$  defined in Assumption 1, and where  $C$  is a universal constant.

The proof of this theorem is highly technical, thus, we defer it to Appendix F.

## 6 EXPERIMENTS

Here we present preliminary experiments showing the advantages of our algorithm. We also present empirical evidence showing that the procedure proposed in Sun et al. (2014) to estimate valuations from bids is incorrect. In contrast, our density estimation algorithm correctly recovers valuations from bids in equilibrium.

### 6.1 SETUP

Let  $F_1$  and  $F_2$  denote the distributions of two truncated log-normal random variables with parameters  $\mu_1 = \log(.5)$ ,  $\sigma_1 = .8$  and  $\mu_2 = \log(2)$ ,  $\sigma = .1$ ; the mixture parameter was set to  $1/2$ . Here,  $F_1$  is truncated to have support in  $[0, 1.5]$  and the support of  $F_2 = [0, 2.5]$ . We consider a GSP with  $N = 4$  advertisers with  $S = 3$  slots and position factors  $c_1 = 1$ ,  $c_2 = .45$  and  $c_3 = 1$ . Based on the results of Section 5 we estimate the bidding function  $\beta$  with a sample of 2000 points and we show its plot in Figure 4. We proceed to evaluate the method proposed by Sun et al. (2014) for

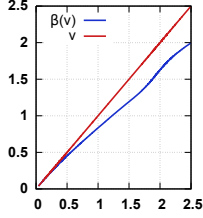


Figure 4: Bidding function for our experiments in blue and identity function in red.

recovering advertisers valuations from bids in equilibrium. The assumption made by the authors is that the advertisers play a SNE in which case valuations can be inferred by solving a simple system of inequalities defining the SNE (Varian, 2007). Since the authors do not specify which SNE the advertisers are playing we select the one that solves the SNE conditions with equality.

We generated a sample  $\mathcal{S}$  consisting of  $n = 300$  i.i.d. outcomes of our simulated auction. Since  $N = 4$ , the effective size of this sample is of 1200 points. We generated the outcome bid vectors  $\mathbf{b}_1, \dots, \mathbf{b}_n$  by using the equilibrium bidding function  $\beta$ . Assuming that the bids constitute a SNE we estimated the valuations and Figure 5 shows an histogram of the original sample as well as the histogram of the estimated valuations. It is clear from this figure that this procedure does not accurately recover the distribution of the valuations. By contrast, the histogram of the estimated valuations using our density estimation algorithm is shown in Figure 5(c). The kernel function used by our algorithm was a triangular kernel given by  $K(u) = (1 - |u|)\mathbb{1}_{|u| \leq 1}$ . Following the experimental setup of Guerre et al. (2000) the bandwidth  $h$  was set to  $h = 1.06\hat{\sigma}n^{1/5}$ , where  $\hat{\sigma}$  denotes the standard deviation of the sample of bids.

Finally, we use both our density estimation algorithm and discriminative learning algorithm to infer the optimal value of  $r$ . To test our algorithm we generated a test sample of size  $n = 500$  with the procedure previously described. The results are shown in Table 1.

| Density estimation | Discriminative  |
|--------------------|-----------------|
| $1.42 \pm 0.02$    | $1.85 \pm 0.02$ |

Table 1: Mean revenue for our two algorithms.

## 7 CONCLUSION

We proposed and analyzed two algorithms for learning optimal reserve prices for generalized second price auctions. Our first algorithm is based on density estimation and therefore suffers from the standard problems associated with this family of algorithms. Furthermore, this algorithm is only well defined when bidders play in equilibrium. Our second algorithm is novel and is based on learning theory

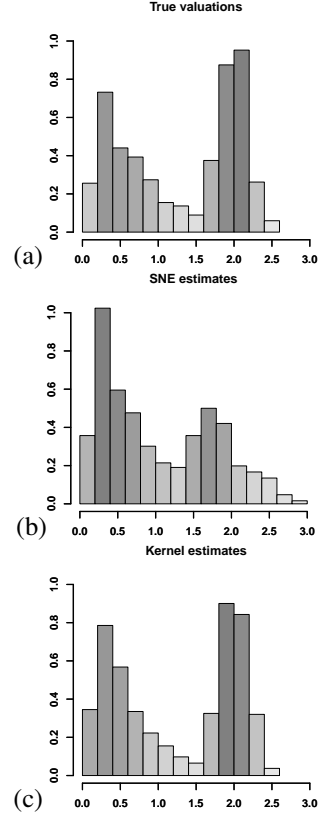


Figure 5: Comparison of methods for estimating valuations from bids. (a) Histogram of true valuations. (b) Valuations estimated under the SNE assumption. (c) Density estimation algorithm.

guarantees. We show that the algorithm admits an efficient  $O(nS \log(nS))$  implementation. Furthermore, our theoretical guarantees are more favorable than those presented for the previous algorithm of Sun et al. (2014). Moreover, even though it is necessary for advertisers to play in equilibrium for our algorithm to converge to optimality, when bidders do not play an equilibrium, our algorithm is still well defined and minimizes a quantity of interest albeit over a smaller set. We also presented preliminary experimental results showing the advantages of our algorithm. To our knowledge, this is the first attempt to apply learning algorithms to the problem of reserve price selection in GSP auctions. We believe that the use of learning algorithms in revenue optimization is crucial and that this work may preface a rich research agenda including extensions of this work to a general learning setup where auctions and advertisers are represented by features. Additionally, in our analysis, we considered two different ranking rules. It would be interesting to combine the algorithm of Zhu et al. (2009) with this work to learn both a ranking rule and an optimal reserve price. Finally, we provided the first analysis of convergence of bidding functions in an empirical equilibrium to the true bidding function. This result on its own is of great importance as it justifies the common assumption of advertisers playing in a Bayes-Nash equilibrium.

## References

- Baker, C. T. (1977). *The numerical treatment of integral equations*. Clarendon press.
- Börger, T., I. Cox, M. Pesendorfer, and V. Petricek (2013). Equilibrium bids in sponsored search auctions: Theory and evidence. *American Economic Journal: Microeconomics* 5(4), 163–87.
- Cesa-Bianchi, N., C. Gentile, and Y. Mansour (2013). Regret minimization for reserve prices in second-price auctions. In *Proceedings of SODA 2013*, pp. 1190–1204.
- Edelman, B., M. Ostrovsky, and M. Schwarz (2005). Internet advertising and the generalized second price auction: Selling billions of dollars worth of keywords. *American Economic Review* 97.
- Edelman, B. and M. Schwarz (2010). Optimal auction design and equilibrium selection in sponsored search auctions. *American Economic Review* 100(2), 597–602.
- Gibbons, R. (1992). *Game theory for applied economists*. Princeton University Press.
- Gomes, R. and K. S. Sweeney (2014). Bayes-Nash equilibria of the generalized second-price auction. *Games and Economic Behavior* 86, 421–437.
- Guerre, E., I. Perrigne, and Q. Vuong (2000). Optimal non-parametric estimation of first-price auctions. *Econometrica* 68(3), 525–574.
- He, D., W. Chen, L. Wang, and T. Liu (2014). A game-theoretic machine learning approach for revenue maximization in sponsored search. *CoRR abs/1406.0728*.
- Kress, R., V. Maz'ya, and V. Kozlov (1989). *Linear integral equations*, Volume 82. Springer.
- Lahaie, S. and D. M. Pennock (2007). Revenue analysis of a family of ranking rules for keyword auctions. In *Proceedings of ACM EC*, pp. 50–56.
- Ledoux, M. and M. Talagrand (2011). *Probability in Banach spaces*. Classics in Mathematics. Berlin: Springer-Verlag. Isoperimetry and processes, Reprint of the 1991 edition.
- Linz, P. (1985). *Analytical and numerical methods for Volterra equations*, Volume 7. SIAM.
- Lucier, B., R. P. Leme, and É. Tardos (2012). On revenue in the generalized second price auction. In *Proceedings of WWW*, pp. 361–370.
- Milgrom, P. and I. Segal (2002). Envelope theorems for arbitrary choice sets. *Econometrica* 70(2), 583–601.
- Mohri, M. and A. M. Medina (2014). Learning theory and algorithms for revenue optimization in second price auctions with reserve. In *Proceedings of ICML*, pp. 262–270.
- Myerson, R. (1981). Optimal auction design. *Mathematics of operations research* 6(1), 58–73.
- Ostrovsky, M. and M. Schwarz (2011). Reserve prices in internet advertising auctions: a field experiment. In *Proceedings of ACM EC*, pp. 59–60.
- Pardoe, D., P. Stone, M. Saar-Tsechansky, and K. Tomak (2005). Adaptive auctions: Learning to adjust to bidders. In *Proceedings of WITS 2005*.
- Qin, T., W. Chen, and T. Liu (2014). Sponsored search auctions: Recent advances and future directions. *ACM TIST* 5(4), 60.
- Sun, Y., Y. Zhou, and X. Deng (2014). Optimal reserve prices in weighted GSP auctions. *Electronic Commerce Research and Applications* 13(3), 178–187.
- Thompson, D. R. M. and K. Leyton-Brown (2013). Revenue optimization in the generalized second-price auction. In *Proceedings of ACM EC*, pp. 837–852.
- Varian, H. R. (2007, December). Position auctions. *International Journal of Industrial Organization* 25(6), 1163–1178.
- Vickrey, W. (1961). Counterspeculation, auctions, and competitive sealed tenders. *The Journal of finance* 16(1), 8–37.
- Vorobeychik, Y. (2009). Simulation-based analysis of keyword auctions. *SIGecom Exchanges* 8(1).
- Zhu, Y., G. Wang, J. Yang, D. Wang, J. Yan, J. Hu, and Z. Chen (2009). Optimizing search engine revenue in sponsored search. In *Proceedings of ACM SIGIR*, pp. 588–595.

---

# Polynomial-time algorithm for learning optimal tree-augmented dynamic Bayesian networks

---

José L. Monteiro  
IDMEC

Instituto Superior Técnico  
Universidade de Lisboa

jose.libano.monteiro@tecnico.ulisboa.pt

Susana Vinga  
IDMEC

Instituto Superior Técnico  
Universidade de Lisboa

susanavinga@tecnico.ulisboa.pt

Alexandra M. Carvalho

Instituto de Telecomunicações  
Instituto Superior Técnico  
Universidade de Lisboa

alexandra.carvalho@tecnico.ulisboa.pt

## Abstract

The identification of conditional dependences in longitudinal data is provided through structure learning of dynamic Bayesian networks (DBN). Several methods for DBN learning are concerned with identifying inter-slice dependences, but often disregard the intra-slice connectivity. We propose an algorithm that jointly finds the optimal inter and intra time-slice connectivity in a transition network. The search space is constrained to a class of networks designated by tree-augmented DBN, leading to polynomial time complexity. We assess the effectiveness of the algorithm on simulated data and compare the results to those obtained by a state of the art DBN learning implementation, showing that the proposed algorithm performs very well throughout the different experiments. Further experimental validation is made on real data, by identifying non-stationary gene regulatory networks of *Drosophila melanogaster*.

## 1 INTRODUCTION

Longitudinal data, also known as multivariate time series in the machine learning community, are obtained by conducting repeated measurements on a set of individuals. They arise in several contexts, such as biomedical and clinical studies, socio-economics and meteorology, and provide an opportunity for studying changes over time. In multivariate longitudinal data, each measurement or observation is a vector of variables, whose joint evolution is subject of analysis.

Multivariate longitudinal data can be modelled as a set of  $n$ -dimensional observations of a stochastic process over  $T$  sequential instants of time. The set of observations is expressed as  $\{\mathbf{x}^i[t]\}_{i \in \mathcal{I}, t \in \mathcal{T}}$ , where  $\mathcal{I}$  is the set of individuals being measured and  $\mathcal{T}$  is the set of time indices. Thus,

$\mathbf{x}^i[t] = (x_1^i[t], \dots, x_n^i[t]) \in \mathbb{R}^n$  is a single observation of  $n$  features, made at time  $t$  and referring to the individual  $i$ .

Observations are assumed to result from independent samples of a sequence of probability distributions  $\{\mathbb{P}_{\theta[t]}\}_{t \in \mathcal{T}}$ . While these distributions may be time-variant, they are considered constant across different individuals observed at the same time, such that  $\mathbf{x}^i[t] \sim \mathbb{P}_{\theta[t]}$  for all  $i \in \mathcal{I}$ . If the observations are also identically distributed over time, that is,  $\theta[t] = \theta$  for all  $t \in \mathcal{T}$ , the process is said to have a stationary or time-invariant distribution (henceforth simply referred to as a stationary process).

The identification of conditional independences in data provides an approximation to estimating the underlying probability distribution (Chow and Liu, 1968). Bayesian networks (BN) are a popular machine learning tool for this purpose, being able to represent complex processes that involve uncertainty. Dynamic Bayesian networks (DBN) extend BN in order to model temporal processes and are usually defined according to strong simplifying assumptions (Friedman et al., 1998; Murphy, 2002). A common premise is to consider the first-order Markov property, which states that attributes in time-slice  $t + 1$  only depend on those in time-slice  $t$ , but not on the past trajectory. Another frequent assumption is to consider a stationary process, which may be adequate in some cases, but does not hold in many interesting scenarios.

### 1.1 RELATED WORK

Methods for learning stationary DBN are essentially simple extensions of those employed to learn BN (Murphy, 2002). A common approach consists in defining a scoring function, which measures a network's goodness of fit to training data, and a search procedure to generate networks (Heckerman et al., 1995). In general, obtaining an optimal network is an NP-hard problem, because the search space is super-exponential in the number of attributes (Chickering et al., 1995).

Unlike the case of BN, however, it was recently shown that learning the inter-slice structure of DBN does not have



to be NP-hard (Dojer, 2006). This new hardness result is due to the relaxation of the acyclicity constraint on the transition network, since the unrolled network is always acyclic. In the same article, the author derived a polynomial complexity bound in the number of variables when using the minimum description length (MDL) or the Bayesian Dirichlet equivalence (BDe) scores. Relying on this result, Vinh et al. (2011b) further proposed a polynomial-time algorithm for learning optimal DBN using the mutual information tests (MIT) score.

While there is plenty of literature regarding the process of learning stationary first-order Markov networks, there are only a few references to learning more general classes of DBN. In fact, it was not until recently that some authors started to relax the standard stationarity assumption underlying graphical models. The following paragraphs present a brief review of such realizations.

The problem of model selection, that is, identifying a system for probabilistic inference that is efficient, accurate and informative is discussed in Bilmes (2000). With the purpose of performing classification, the author proposes a class of models where the conditional independences are determined by the values of certain nodes in the graph, instead of being fixed. This is accomplished by extending hidden Markov models (HMM) to include dependences among observations of different time-slices. The idea of a network whose edges can appear and disappear is further explored by other authors in a temporal context to model non-stationary processes.

An extension of the traditional discrete DBN model is defined in Robinson and Hartemink (2010), where an initial network of dependences and a set of incremental changes on its structure are learnt. The authors assume that the process is piecewise-stationary, having the number and times of the transitions (change points) to be estimated a posteriori. Prior knowledge regarding both the initial structure and the evolutionary behaviour of the network can be incorporated. By considering conjugate Dirichlet priors on the parameters, which are assumed to have a multinomial distribution, the marginal likelihood is computed exactly, resulting in the BDe metric. The authors extend this metric to incorporate the changes introduced by their new model.

The same approach is considered in Dondelinger et al. (2010), but concerning continuous data. The authors also differentiate the penalties for adding and removing edges in a network and allow different nodes to have distinct penalty terms, instead of a single hyperparameter for penalizing disparities between structures.

In more recent work, Grzegorzczak and Husmeier (2012) argue that there should be a trade-off between the often unrealistic stationarity assumption, modelled with constant parameters, and the opposite case of complete parameter independence over time, ignoring the evolutionary aspect

of the process. They acknowledge, however, that the latter case, which is considered in Dondelinger et al. (2010) and Robinson and Hartemink (2010), has the advantage of allowing the computation of the marginal likelihood in closed form. The authors introduce a scheme for coupling the parameters along time-slices, although keeping the network structure fixed.

Regarding undirected graphical models, Kolar et al. (2010) propose two methods for estimating the underlying time-varying networks of a stochastic process. They model each network as a Markov random field (MRF) with binary nodes. The first method assumes that the parameters change smoothly over time whereas the second considers piecewise constant parameters with abrupt changes. In both approaches, the estimator for the parameters is the result of a  $l_1$ -regularized convex optimization problem. These methods, however, only capture pairwise undirected relations between binary variables, resulting in a model which is far from being generally applicable.

## 1.2 OUR APPROACH

The algorithm we propose falls under the search and score paradigm. Many software implementations for learning DBN are concerned with identifying inter-slice dependences, but disregard the intra-slice connectivity or assume it is given by some prior network and kept fixed over time (Dojer et al., 2013; Murphy, 2001; Vinh et al., 2011a). We instead suggest an algorithm that simultaneously learns all these dependences.

As a consequence of considering intra-slice edges in the proposed algorithm, the relaxation of the acyclicity constraint proposed in Dojer (2006) no longer applies, and obtaining an optimal network becomes NP-hard. We approach this problem by limiting the search space to tree-augmented networks, that is, networks whose attributes have at most one parent in the same time-slice. This restriction does not prevent an attribute to have several parents from preceding slices, and also accounts for the algorithm's polynomial time complexity in the number of attributes. Moreover, even though tree structures appear to be a strong constraint, they have been shown to produce very good results in classification tasks, namely within the tree augmented naive Bayes (TAN) method (Friedman et al., 1997).

The remaining of the paper is organized as follows. Section 2 formally defines BN, provides a theoretical overview on learning this class of networks and introduces DBN by extension. Section 3 describes the proposed DBN structure learning algorithm and analyses its time complexity. Section 4 assesses its performance on simulated data and real data. Section 5 presents the conclusions of this work.

## 2 THEORETICAL BACKGROUND

A BN is a graphical representation of a joint probability distribution over a set of random variables (Pearl, 1988). It is defined as a triple  $B = (\mathbf{X}, G, \theta)$ , where:

- $\mathbf{X} = (X_1, \dots, X_n)$  is a random vector. Discrete random variables with a finite domain are considered;
- $G = (\mathbf{X}, E)$  is a directed acyclic graph (DAG) whose nodes are the elements of  $\mathbf{X}$  and edges  $E$  specify conditional dependences between the variables: each  $X_i$  is independent of its non-descendants given its parents  $\mathbf{pa}(X_i)$  in  $G$ ;
- $\theta = \{\theta_{ijk}\}$  is a set of parameters, specifying the local probability distributions of the network via

$$\theta_{ijk} = P_B(X_i = x_{ik} \mid \mathbf{pa}(X_i) = w_{ij}), \quad (1)$$

where  $i \in \{1, \dots, n\}$ ,  $j \in \{1, \dots, q_i\}$  and  $k \in \{1, \dots, r_i\}$ .  $r_i$  is the number of discrete states of  $X_i$ . The set of possible configurations of  $\mathbf{pa}(X_i)$ , i.e., the set of different combinations of values that the parents of  $X_i$  can take, is denoted by  $\{w_{i1}, \dots, w_{iq_i}\}$ , where  $q_i = \prod_{X_j \in \mathbf{pa}(X_i)} r_j$  is the number of all possible configurations.

A BN  $B$  defines a joint probability distribution over  $\mathbf{X}$ :

$$P_B(X_1, \dots, X_n) = \prod_{i=1}^n P_B(X_i \mid \mathbf{pa}(X_i)). \quad (2)$$

The problem of learning a BN, given a dataset comprising instances of  $\mathbf{X}$ , can be stated as finding the structure (DAG) and parameters that best match the training data. When measuring the goodness of fit of a network  $B$  to data  $D$  by means of a scoring function  $\phi$ , learning a BN consists of maximizing  $\phi(B, D)$  over the space of all networks with  $n$  attributes.

A decomposable scoring function can be expressed as a sum of local terms, each depending only on a node and its parents:

$$\phi(B, D) = \sum_{i=1}^n \phi_i(\mathbf{pa}(X_i), D). \quad (3)$$

Decomposability simplifies the process of calculating scores and provides an efficient way of evaluating incremental changes on a network.

The log-likelihood (LL) is a decomposable score which favours networks that are more likely to have generated the data. For a fixed structure  $G$ , assuming an underlying multinomial distribution, the network parameters are determined by maximum-likelihood estimation (MLE):

$$\{\hat{\theta}_{ijk} = \hat{P}_D(X_i = x_{ik} \mid \mathbf{pa}(X_i) = w_{ij}) = \frac{N_{ijk}}{N_{ij}}\} \quad (4)$$

where  $\hat{P}_D$  is the distribution induced by the observed frequency estimates,  $N_{ijk}$  is the number of instances where  $X_i$  takes its  $k$ -th value  $x_{ik}$  and the variables in  $\mathbf{pa}(X_i)$  take their  $j$ -th configuration  $w_{ij}$ , and  $N_{ij}$  is the number of instances where the variables in  $\mathbf{pa}(X_i)$  take their  $j$ -th configuration  $w_{ij}$  notwithstanding the value of  $X_i$ . Since the parameters are unambiguously found for a fixed network structure, the LL criterion depends only on the network  $G$ . Taking into account Eq. (2), and assuming that instances of  $D$  are independent and identically distributed (i.i.d.), the LL scoring function is expressed (Heckerman et al., 1995) as

$$\begin{aligned} \phi_{LL}(B, D) &= \log P(D \mid B) \\ &= \sum_{i=1}^n \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} N_{ijk} \log \frac{N_{ijk}}{N_{ij}}. \end{aligned} \quad (5)$$

The minimum description length (MDL) score is an extension of the LL criterion, including a term for penalizing complex structures:

$$\phi_{MDL}(B, D) = \phi_{LL}(B, D) - \frac{1}{2} \log(N) |B|, \quad (6)$$

where  $N$  is the number of samples in  $D$  and  $|B|$  denotes the number of parameters of the network, given by:

$$|B| = \sum_{i=1}^n (r_i - 1) q_i. \quad (7)$$

While a BN defines a joint probability distribution over a fixed set of variables, a DBN extends this representation to model temporal processes (Friedman et al., 1998). Let  $\mathbf{X} = (X_1, \dots, X_n)$  be a random vector, composed by the attributes that are changed by some process. Furthermore, let  $\mathbf{X}[t] = (X_1[t], \dots, X_n[t])$  denote the instantiation of the attributes at discrete time  $t \in \mathbb{N}$ . A DBN encodes the joint probability distributions over all possible trajectories of a process.

The first-order Markov property states that future values only depend on present ones, not on the past trajectory, s.t.  $P(\mathbf{X}[t+1] \mid \mathbf{X}[0] \cup \dots \cup \mathbf{X}[t]) = P(\mathbf{X}[t+1] \mid \mathbf{X}[t])$ . A relaxation of this assumption is the higher-order Markov property, where nodes can have dependences on an arbitrary (but fixed) number of previous time-slices.

A non-stationary first-order Markov DBN describing a temporal process over  $T$  time-slices consists of:

- a prior network  $B^0$ , which specifies a distribution over the initial states  $\mathbf{X}[0]$ ;
- a set of transition networks  $B_t^{t+1}$  over the variables  $\mathbf{X}[t] \cup \mathbf{X}[t+1]$ , specifying the state transition probabilities, for  $0 \leq t < T$ .

A stationary network contains only one prior network and one transition network, being the latter unrolled over time.

Learning DBN typically refers to the transition network(s), as learning the prior network can be done directly using BN methods. Learning a transition network has the additional requirement that edges between slices must flow forward in time.

### 3 PROPOSED METHOD

The proposed algorithm is based on learning tree-like Bayesian networks. The Chow-Liu algorithm finds a tree with maximum mutual information (Chow and Liu, 1968), or, equivalently, a tree with maximum LL score. The algorithm works as follows: (i) a complete undirected graph weighted with the mutual information between each pair of nodes is built; (ii) an undirected spanning tree is extracted; and (iii) the optimal branching is retrieved by choosing an arbitrary node as the tree root and then setting the direction of all edges to be outward from it.

It was shown that the Chow-Liu algorithm can be adapted to use any decomposable scoring criterion  $\phi$  (Heckerman et al., 1995). In this case, the weight of an edge  $X_j \rightarrow X_i$  is assigned as  $\phi_i(\{X_j\}, D) - \phi_i(\{\}, D)$ , expressing the contribution of the edge, as measured by  $\phi$ , to the total network score. If  $\phi$  is score-equivalent, the weights of the edges  $X_j \rightarrow X_i$  and  $X_i \rightarrow X_j$  are the same, and so an undirected spanning tree is enough to retrieve the optimal branching. However, if the score is not score-equivalent, Edmond’s algorithm (Edmonds, 1967) needs to be used to find the maximum branching from a complete directed weighted graph (Heckerman et al., 1995).

In the temporal domain, nodes in  $\mathbf{X}[t + 1]$  can also have parents from previous time-slices. Our approach for learning DBN jointly learns inter and intra time-slice dependences. We propose to learn a tree network structure for the intra-slice dependences while limiting the number of parents from the preceding time-slices. That is, for each node in the current time-slice  $t + 1$  we allow one parent from the same time-slice (with the exception of the root node) and at most  $p$  parents from the preceding time-slices. We call the resulting transition network structure a tree-augmented DBN (tDBN). As we shall see next, the weight of an edge  $X_j[t + 1] \rightarrow X_i[t + 1]$  will account for the contribution of inter and intra-slice parents simultaneously. Therefore, due to inter-slice parents, the weights  $X_j[t + 1] \rightarrow X_i[t + 1]$  and  $X_i[t + 1] \rightarrow X_j[t + 1]$  are, in general, not the same. This forces us to resort to Edmond’s algorithm.

#### 3.1 OPTIMAL TREE-AUGMENTED DBN STRUCTURE LEARNING

Considering, for the sake of simplicity, the first-order Markov DBN paradigm, parents from the past can only be

long to the preceding slice. Let  $\mathcal{P}_{\leq p}(\mathbf{X}[t])$  be the set of subsets of  $\mathbf{X}[t]$  of cardinality less than or equal to  $p$ . If a node in  $\mathbf{X}[t + 1]$  is limited to having at most  $p$  parents from the past, its set of parents must belong to  $\mathcal{P}_{\leq p}(\mathbf{X}[t])$ . The optimal tDBN structure learning algorithm proceeds as follows.

First, for each node  $X_i[t + 1] \in \mathbf{X}[t + 1]$ , the best score and the set of parents  $\mathbf{X}_{ps}[t]$  in  $\mathcal{P}_{\leq p}(\mathbf{X}[t])$  that maximizes it are found. This optimization is formally expressed as

$$s_i = \max_{\mathbf{X}_{ps}[t] \in \mathcal{P}_{\leq p}(\mathbf{X}[t])} \phi_i(\mathbf{X}_{ps}[t], D_t^{t+1}), \quad (8)$$

where  $\phi_i$  denotes a local term of a decomposable scoring function  $\phi$  and  $D_t^{t+1}$  is the subset of observations of  $D$  concerning the time transition  $t \rightarrow t + 1$ . Then, also allowing one parent from the current time-slice, for each edge  $X_j[t + 1] \rightarrow X_i[t + 1]$ , the best score and the set of parents from the past that maximizes it are also found:

$$s_{ij} = \max_{\mathbf{X}_{ps}[t] \in \mathcal{P}_{\leq p}(\mathbf{X}[t])} \phi_i(\mathbf{X}_{ps}[t] \cup \{X_j[t + 1]\}, D_t^{t+1}). \quad (9)$$

A complete directed graph with nodes in  $\mathbf{X}[t + 1]$  is built, being the weight of each edge  $X_j[t + 1] \rightarrow X_i[t + 1]$  assigned as

$$e_{ij} = s_{ij} - s_i, \quad (10)$$

which expresses the gain in the total network score by including  $X_j[t + 1]$  as a parent of  $X_i[t + 1]$ , as opposed to leaving  $X_i[t + 1]$  only with parents in  $\mathbf{X}[t]$ . In general,  $e_{ij} \neq e_{ji}$ , and therefore a directed spanning tree must be found. Thus, to obtain the  $t \rightarrow t + 1$  transition network structure, the Edmonds’ algorithm for finding a maximum branching (Edmonds, 1967) is applied. The resulting directed tree immediately provides the network intra-slice connectivity (in  $t + 1$ ). In addition, for all the nodes except the root, their set of parents from the preceding time-slice is the solution for the optimization problem in Eq. (9). Similarly, the root node’s parents are given by the solution for the problem in Eq. (8).

The described procedure can jointly obtain the intra and inter-slice connectivity in a transition network. By repeatedly applying it to all the available time transitions, it is possible to retrieve the structure of a tree-augmented non-stationary first-order Markov DBN. A global view of this method is presented in Algorithm 1.

**Theorem 1.** *Algorithm 1 finds an optimal tDBN under a given decomposable scoring function.*

*Proof.* Let  $B$  be an optimal tDBN and  $T$  be the tree structure of  $B$  accounting only for the intra-slice dependences. Due to the optimality of  $B$ , its overall score is equal to

$$s_R + \sum_{X_j[t+1] \rightarrow X_i[t+1] \in T} s_{ij},$$

---

**Algorithm 1:** Optimal non-stationary first-order Markov tDBN structure learning

---

**Input:**  $\mathbf{X}$ : the set of network attributes;  
 $D$ : dataset of longitudinal observations over  $T$  time-slices;  
 $\phi$ : a decomposable scoring function

**Output:** A tree-augmented DBN structure

- 1 **For each** transition  $t \rightarrow t + 1$  :
  - 2     Build a complete directed graph in  $\mathbf{X}[t + 1]$
  - 3     Calculate the weight of all edges and the optimal set of parents of all nodes (Algorithm 2)
  - 4     Apply a maximum branching algorithm
  - 5     Extract transition  $t \rightarrow t + 1$  network using the maximum branching and the optimal set of parents calculated in Algorithm 2
  - 6 Collect transition networks to obtain DBN structure
- 

according to Eq. (8) and Eq. (9), where  $X_R$  is the root node of  $T$ .

Consider the constant  $K = \sum_i s_i$ . Finding an optimal tDBN for a given score  $\phi$  is the same as finding the optimal tDBN up to the constant  $K$ ; note that  $s_i$  does not depend on the structure  $T$ , nor  $B$ . By subtracting  $K$  to the score of  $B$  we obtain

$$\sum_{X_j[t+1] \rightarrow X_i[t+1] \in T} e_{ij},$$

according to Eq. (10). Observe that an optimal branching of the complete directed graph, where each edge  $X_j[t + 1] \rightarrow X_i[t + 1]$  is weighted with  $e_{ij}$ , is precisely  $T$ . Therefore, due to the soundness of Edmonds' algorithm, the output of Algorithm 1 is  $T$ , from which  $B$  can be recovered.  $\square$

### 3.2 COMPLEXITY ANALYSIS

The derivation of a complexity bound on the running time of Algorithm 1 is presented in the following.

**Theorem 2.** *The time complexity of Algorithm 1 is polynomial in the number of attributes  $n$ , linear in the the number of observations  $N$ , and exponential in the number of parents  $p$ .*

*Proof.* For each transition, the step of determining the edge weights and optimal sets of parents takes the most number of operations and determines the algorithm's growth rate. The iterative process starting at line 11 in Algorithm 2 is the most expensive overall. It calculates the weights for all edges in a complete graph with  $n$  nodes, which requires  $\mathcal{O}(n^2)$  iterations. For any edge, a score is evaluated for each possible set of parents in the preceding time-slice. The total number of parent sets is given by:

$$|\mathcal{P}_{\leq p}(\mathbf{X}[t])| = \sum_{i=1}^p \binom{n}{i} < \sum_{i=1}^p n^i \in \mathcal{O}(n^p). \quad (11)$$

---

**Algorithm 2:** Determining edge weights and optimal sets of parents (first-order Markov)

---

**Input:**  $\mathbf{X}[t], \mathbf{X}[t + 1]$ : sets of  $n$  nodes from two adjacent time-slices;  
 $p$ : upper bound on the number of parents from time-slice  $t$ ;  
 $D_t^{t+1}$ : dataset of observations concerning the time transition  $t \rightarrow t + 1$ ;

$\phi_{\text{child}[t+1]}(\text{parents}, \text{dataset})$ : a local term of  $\phi$

**Output:**  $E_{[n \times n]}$ : edge weights matrix;  
 $\text{parentsPastSlice}_{[n]}$ : optimal set of parents from time-slice  $t$ ;  
 $\text{parentsAllSlices}_{[n \times n]}$ : optimal set of parents from time-slices  $t$  and  $t + 1$

- 1  $\text{allParentSets} \leftarrow \mathcal{P}_{\leq p}(\mathbf{X}[t])$
  - 2 **For each**  $X_i[t + 1]$  :
  - 3      $\text{bestScore} \leftarrow -\infty$
  - 4     **For each**  $\mathbf{X}_{ps}[t] \in \text{allParentSets}$  :
  - 5          $\text{currentScore} \leftarrow \phi_i(\mathbf{X}_{ps}[t], D_t^{t+1})$
  - 6         **If**  $\text{bestScore} < \text{currentScore}$  :
  - 7              $\text{bestScore} \leftarrow \text{currentScore}$
  - 8              $\text{parentsPastSlice}_i \leftarrow \mathbf{X}_{ps}[t]$
  - 9     **For each**  $X_j[t + 1]$  :
  - 10          $E_{ij} \leftarrow -\text{bestScore}$
  - 11 **For each**  $X_i[t + 1]$  :
  - 12     **For each**  $X_j[t + 1]$  :
  - 13          $\text{bestScore} \leftarrow -\infty$
  - 14         **For each**  $\mathbf{X}_{ps}[t] \in \text{allParentSets}$  :
  - 15              $\text{currentScore}$
  - 16              $\leftarrow \phi_i(\mathbf{X}_{ps}[t] \cup \{X_j[t + 1]\}, D_t^{t+1})$
  - 17             **If**  $\text{bestScore} < \text{currentScore}$  :
  - 18                  $\text{bestScore} \leftarrow \text{currentScore}$
  - 19                  $\text{parentsAllSlices}_{ij} \leftarrow \mathbf{X}_{ps}[t]$
  - 20              $E_{ij} \leftarrow E_{ij} + \text{bestScore}$
- 

For calculating each score, all different network configurations must be considered. Assuming that  $r$  is the maximum number of discrete states a variable can take, and that a variable  $X_i[t + 1]$  has  $p + 1$  parents (one in  $\mathbf{X}[t + 1]$  and  $p$  in  $\mathbf{X}[t]$ ), there are  $\mathcal{O}(r^{p+2})$  different configurations. Each configuration needs to be counted over a dataset containing  $|D_t^{t+1}|$  observations, which can be stored in a  $|D_t^{t+1}| \times 2n$  sized matrix, thus requiring  $\mathcal{O}(|D_t^{t+1}|n)$  comparisons. Taking into account all the internal loops, the complexity of the outer cycle is  $\mathcal{O}(n^{p+3} r^{p+2} |D_t^{t+1}|)$ .

The efficient implementation of Edmonds' algorithm described in (Tarjan, 1977) has quadratic complexity in the number of nodes, hence being irrelevant to the overall bound. Algorithm 1, which learns a network structure for each of  $T$  time transitions, admits a worst-case complexity of  $\mathcal{O}(n^{p+3} r^{p+2} N)$ , where  $N = |D| = \sum_{t=0}^{T-1} |D_t^{t+1}|$ .  $\square$

### 3.3 EXTENSIONS TO STATIONARITY AND HIGHER-ORDER MARKOV

Algorithm 1 was presented in its non-stationary first-order Markov form. Nonetheless, adaptations for stationary or higher-order Markov processes are trivial and can be concisely described.

A stationary version of Algorithm 1 does not contain the for loop starting at line 1, because only one iteration is needed to find one transition network. In Algorithm 2, the entire dataset  $D$  is used in each score evaluation. Overall, since the number of examinations of each observation for learning the DBN structure is the same as in the non-stationary version, the time complexity remains the same, that is,  $\mathcal{O}(n^{p+3} r^{p+2} N)$ .

In a  $m$ -th-order Markov version of tDBN, regardless of process stationarity, nodes are allowed to have parents from  $m$  previous time-slices. Therefore, in Algorithm 2,  $\text{allParentSets} \leftarrow \mathcal{P}_{\leq p}(\mathbf{X}[t-m+1] \cup \dots \cup \mathbf{X}[t])$  and  $D_t^{t+1}$  is used instead of  $D_t^{t+1}$ . These changes worsen the algorithm’s time complexity, which can be roughly approximated by  $\mathcal{O}((nm)^{p+3} r^{p+2} N)$ .

## 4 EXPERIMENTAL RESULTS

In this section we describe the methodology used for evaluating the optimal tDBN learning algorithm and present the obtained results, in terms of speed and accuracy. Our algorithm was implemented in Java using an object-oriented approach and released under a free-software license<sup>1</sup>. Simulated data was first considered to evaluate the simpler stationary first-order Markov tDBN. As very good results were achieved, further assessment on real data with a non-stationary first-order Markov tDBN was used to learn time-varying gene regulatory networks from gene expression data of *Drosophila melanogaster*.

### 4.1 SIMULATED DATA

In the first set of experiments comprising simulated data, Banjo (et al., 2005), a state of the art DBN learning tool, was employed besides the tDBN learning algorithm for comparative purposes. Banjo was chosen for being able to also learn the intra-slice connectivity, as opposed to most DBN learning implementations. Throughout the experiments, an implementation’s ability to recover a known network structure was measured. This was accomplished by specifying a DBN (both its structure and parameters), sampling the network to generate observations and inputting the produced datasets to each implementation, in order to learn the underlying structure. The original and recovered networks were then compared by evaluating the precision

<sup>1</sup>Available at <http://josemonteiro.github.io/tDBN/> where additional experimental results are also provided.

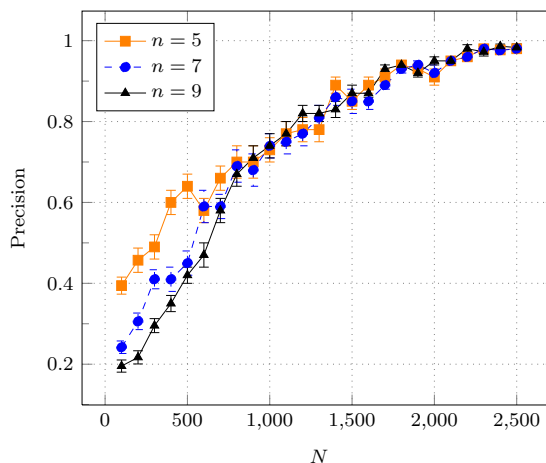


Figure 1: Plot of the precision values achieved by the tDBN+LL learning algorithm for different values of input observations  $N$ . Three lines are shown, corresponding to complete tree-augmented networks with different number of attributes  $n$ , each attribute taking  $r = 8$  different states and having  $p = 2$  parents from the previous time-slice. Each point results from averaging the precision over 25 sampled datasets, with error bars denoting standard errors. Precision generally increases with  $N$  for every choice of  $n$ , attaining a plateau with  $N > 2000$ .

and recall metrics. In addition, to provide a unified score combining the two previous metrics, the  $F$ -measure was also calculated.

Because many methods only learn the inter-slice connectivity of DBN, Table 1 present the metrics taking into account (i) only the inter-slice edges, and (ii) all edges. The results in Table 1 are displayed as average statistics over 5 runs. Precision (Pre), recall (Rec) and  $F$ -measure ( $F_1$ ) values are presented as percentages, running time is in seconds;  $n$  is the number of network attributes,  $p$  is the number of parents from the preceding time-slice,  $r$  is the number of states of all attributes, and  $N$  is the number of observations. Values in bold correspond to the highest  $F$ -measure score in groups (i) or (ii); that is, one concerning the recovery of the inter-slice edges only and another concerning the recovery of all edges. Additional results and details concerning other experiments employing the tDBN learning algorithm are shown in Figures 1 and 2.

In the comparative tests, the stationary tDBN learning algorithm was employed using the LL and MDL scores. Banjo’s Markov lag interval was set between 0 and 1 to allow intra-slice edges. Its running time was set to 10 minutes, which was always longer than learning tDBN’s, and simulated annealing was used as search procedure. In all cases, the maximum number of parents was set according to the original network. The experiments were run on an Intel i5-3570 @ 3.40 GHz machine.

Table 1: Comparative structure recovery results on simulated data. Banjo’s running time is not shown, as it was always 600 seconds.

| N   | tDBN+LL        |                |            |                |                |            |      | tDBN+MDL      |               |           |               |               |           |      | Banjo       |        |        |         |        |       |
|---|----------------|----------------|------------|----------------|----------------|------------|------|---------------|---------------|-----------|---------------|---------------|-----------|------|-------------|--------|--------|---------|--------|-------|
|   | Inter-slice    |                |            | Global         |                |            |      | Inter-slice   |               |           | Global        |               |           |      | Inter-slice |        | Global |         |        |       |
|   | Pre            | Rec            | $F_1$      | Pre            | Rec            | $F_1$      | Time | Pre           | Rec           | $F_1$     | Pre           | Rec           | $F_1$     | Time | Pre         | Rec    | $F_1$  | Pre     | Rec    | $F_1$ |
| Complete tree-augmented network ( $n = 20, p = 2, r = 2$ )        |                |                |            |                |                |            |      |               |               |           |               |               |           |      |             |        |        |         |        |       |
| 100   | <b>64 ± 3</b>  | <b>64 ± 3</b>  | <b>64</b>  | <b>61 ± 4</b>  | <b>61 ± 4</b>  | <b>61</b>  | 4    | 75 ± 3        | 54 ± 3        | 63        | 67 ± 3        | 54 ± 3        | 60        | 4    | 98 ± 3      | 17 ± 1 | 29     | 66 ± 9  | 15 ± 1 | 24    |
| 300   | <b>88 ± 2</b>  | <b>88 ± 2</b>  | <b>88</b>  | <b>86 ± 3</b>  | <b>86 ± 3</b>  | <b>86</b>  | 12   | <b>98 ± 2</b> | <b>80 ± 1</b> | <b>88</b> | 90 ± 2        | 79 ± 1        | 84        | 13   | 98 ± 3      | 18 ± 1 | 30     | 54 ± 3  | 20 ± 1 | 29    |
| 700   | <b>97 ± 1</b>  | <b>97 ± 1</b>  | <b>97</b>  | <b>98 ± 1</b>  | <b>98 ± 1</b>  | <b>98</b>  | 28   | 100 ± 0       | 93 ± 0        | 96        | 100 ± 0       | 95 ± 0        | 97        | 29   | 97 ± 3      | 19 ± 1 | 32     | 46 ± 3  | 19 ± 1 | 27    |
| Complete tree-augmented network ( $n = 20, p = 2, r = 5$ )        |                |                |            |                |                |            |      |               |               |           |               |               |           |      |             |        |        |         |        |       |
| 100   | <b>15 ± 2</b>  | <b>15 ± 2</b>  | <b>15</b>  | <b>13 ± 2</b>  | <b>13 ± 2</b>  | <b>13</b>  | 68   | 21 ± 2        | 11 ± 1        | 14        | <b>16 ± 1</b> | <b>11 ± 1</b> | <b>13</b> | 68   | -           | -      | -      | -       | -      | -     |
| 300   | <b>84 ± 3</b>  | <b>84 ± 3</b>  | <b>84</b>  | <b>83 ± 3</b>  | <b>83 ± 3</b>  | <b>83</b>  | 209  | 51 ± 5        | 26 ± 2        | 34        | 45 ± 5        | 29 ± 3        | 35        | 213  | -           | -      | -      | -       | -      | -     |
| 700   | <b>100 ± 0</b> | <b>100 ± 0</b> | <b>100</b> | <b>100 ± 0</b> | <b>100 ± 0</b> | <b>100</b> | 491  | 97 ± 2        | 49 ± 1        | 65        | 96 ± 2        | 64 ± 1        | 77        | 489  | 100 ± 0     | 3 ± 0  | 6      | 100 ± 0 | 2 ± 0  | 4     |
| Incomplete tree-augmented network ( $n = 20, \max p = 3, r = 2$ ) |                |                |            |                |                |            |      |               |               |           |               |               |           |      |             |        |        |         |        |       |
| 100   | 39 ± 1         | 81 ± 3         | 53         | 43 ± 0         | 70 ± 1         | 53         | 44   | <b>73 ± 2</b> | <b>67 ± 2</b> | <b>70</b> | <b>70 ± 2</b> | <b>66 ± 3</b> | <b>68</b> | 46   | 96 ± 4      | 17 ± 1 | 29     | 60 ± 14 | 18 ± 3 | 28    |
| 300   | 43 ± 1         | 90 ± 1         | 58         | 53 ± 1         | 87 ± 2         | 66         | 136  | <b>85 ± 1</b> | <b>84 ± 2</b> | <b>84</b> | <b>86 ± 3</b> | <b>85 ± 3</b> | <b>85</b> | 140  | 100 ± 0     | 19 ± 1 | 32     | 46 ± 3  | 23 ± 1 | 31    |
| 700   | 48 ± 0         | 99 ± 1         | 65         | 58 ± 1         | 96 ± 2         | 72         | 330  | <b>90 ± 0</b> | <b>94 ± 1</b> | <b>92</b> | <b>94 ± 0</b> | <b>96 ± 0</b> | <b>95</b> | 326  | 93 ± 7      | 20 ± 2 | 33     | 37 ± 5  | 21 ± 3 | 27    |
| Inter-sliced only network ( $n = 20, p = 2, r = 2$ )              |                |                |            |                |                |            |      |               |               |           |               |               |           |      |             |        |        |         |        |       |
| 100   | <b>63 ± 3</b>  | <b>63 ± 3</b>  | <b>63</b>  | <b>42 ± 2</b>  | <b>63 ± 3</b>  | <b>50</b>  | 4    | 73 ± 2        | 51 ± 3        | 60        | 43 ± 2        | 51 ± 3        | 47        | 4    | 100 ± 0     | 18 ± 1 | 31     | 100 ± 0 | 18 ± 1 | 31    |
| 300   | <b>87 ± 1</b>  | <b>87 ± 1</b>  | <b>87</b>  | <b>59 ± 1</b>  | <b>87 ± 1</b>  | <b>70</b>  | 12   | 91 ± 1        | 79 ± 1        | 85        | 59 ± 1        | 79 ± 1        | 68        | 12   | 100 ± 0     | 18 ± 0 | 31     | 89 ± 11 | 18 ± 0 | 30    |
| 700   | 91 ± 1         | 91 ± 1         | 91         | <b>61 ± 1</b>  | <b>91 ± 1</b>  | <b>73</b>  | 28   | <b>96 ± 1</b> | <b>88 ± 1</b> | <b>92</b> | <b>63 ± 0</b> | <b>88 ± 1</b> | <b>73</b> | 28   | 100 ± 0     | 18 ± 0 | 31     | 100 ± 0 | 18 ± 0 | 31    |

Four different network settings are considered in Table 1. The first two networks are complete tree-augmented DBN, in the sense that each attribute in  $\mathbf{X}[t + 1]$  has exactly  $p$  parents in  $\mathbf{X}[t]$  and at most one parent in  $\mathbf{X}[t + 1]$ . In these settings, the number of edges is always  $n(p + 1) - 1$ . On the other hand, the third network is an incomplete tree-augmented DBN, because the number of parents from the preceding slice is chosen at random between 1 and  $p$ . In the fourth network, there are no edges inside  $\mathbf{X}[t + 1]$ , corresponding to the traditional inter-sliced DBN concept.

According to each network setting, a network was created by randomly generating its structure and parameters using uniform distributions. An experimental group, corresponding to a line in Table 1, consisted of 5 independent datasets with the same number of observations, sampled from one of the generated networks. The presented values result from averaging the performance metrics over the datasets of a group.

From the experimental results in Table 1, it can be seen that the performance of the proposed algorithm consistently increases with  $N$ . In the first setting, the tDBN learning algorithm performs very well with either score, with LL having a slight advantage. This result was expected, since a complete tree-augmented DBN is biased towards tDBN, and LL assures that a necessary and sufficient number of edges is recovered. Banjo obtains a very good inter-slice precision, but only recovers one fifth of the original edges. Notice that Banjo’s global results deteriorate when  $N$  increases, which can be explained by decreasing performance while identifying the intra-slice connectivity.

In the second setting, the tDBN+LL learning algorithm globally outperforms the other implementations. The regularization effect of MDL is observed through lower recall levels, since the number of network parameters increases

with  $r$ . Nevertheless, tDBN+MDL greatly improves with  $N$  and already achieves very high precision for  $N = 700$ . Comparing to the first setting, Banjo is more conservative adding edges, and chooses none for  $N \leq 300$ .

In the third setting, the inter-slice precision of tDBN+LL does not exceed 50%, due to the recovery of exactly  $p$  parents, when the real number can be smaller than  $p$ . In this setting, tDBN+MDL clearly achieves the best performance. The penalizing term in MDL prevents false positive edges from being chosen, resulting in significantly higher precision values compared to LL. Banjo performs like in the first setting, being able to determine the correct parents and thus reaching high precision values with respect to the inter-slice connectivity.

Even though the last setting comprises a network without intra-slice edges, the tDBN learning algorithm performs well with either scoring function. In fact, the inter-slice metrics are comparable to the ones in the first setting. The  $F$ -measure values of LL and MDL are very similar, causing neither score to stand out. The recovery of an intra-slice tree, which is an inherent aspect of tDBN learning procedure, worsens the global performance of the algorithm. In this setting, Banjo obtains an excellent inter-slice result, correctly identifying all the dependences and reporting no false positives. On the other hand, the percentage of retrieved edges is quite low, resulting in unimpressive  $F$ -measure scores.

Overall, the tDBN learning algorithm obtained very good results. While tDBN+LL outperformed in the more complex second setting network, tDBN+MDL showed to be more robust, achieving at least one highest  $F$ -measure score in each setting. Banjo consistently identified high-precision sparse inter-sliced networks, but generally could not recover more than 20% of the existing data depen-

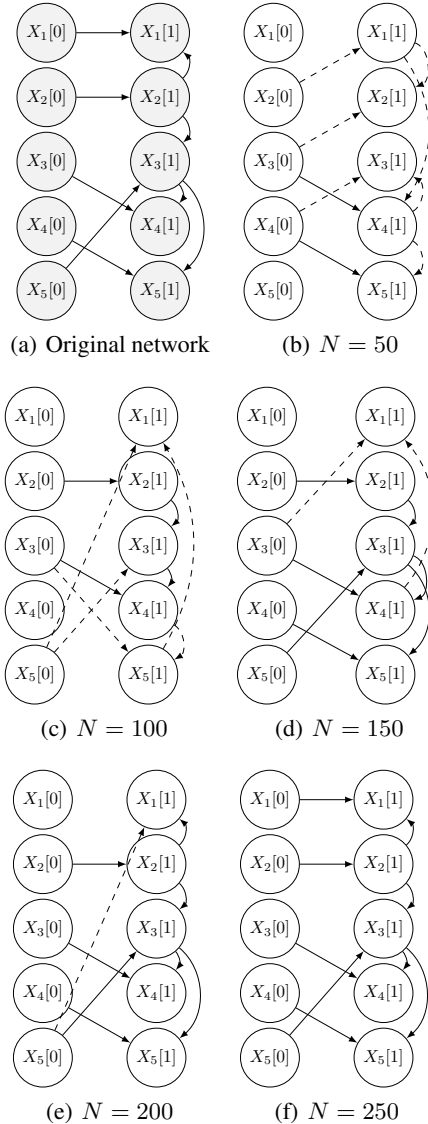


Figure 2: Example of the tDBN+LL learning algorithm’s ability to recover a known network. The original tree-augmented network has  $n = 5$  attributes, each taking  $r = 8$  different states and having one parent from the previous time-slice. Differing in the number of input observations  $N$ , five recovered networks are shown. Dashed edges are not present in the original network but are nevertheless recovered. As  $N$  increases, the recovered network structures become more similar to the original, being identical for  $N = 250$ .

dences.

## 4.2 DROSOPHILA MELANOGASTER DATA

The following experiments consisted in applying the tDBN learning algorithm to identify non-stationary gene regulatory networks of *Drosophila melanogaster*. Arbeitman

et al. (2002) published a dataset containing gene expression measurements of 4028 *Drosophila* genes over 67 time steps, covering the four major stages of morphogenesis: embryo, larva, pupa and adult. Some authors have focused on a small subset of this time series, consisting of eleven genes involved in wing muscle development (Guo et al., 2007; Robinson and Hartemink, 2010; Dondelinger et al., 2013). Consequently, to allow comparison to other methods, the same subset is considered herein. The *Drosophila* gene expression dataset was preprocessed in the same way as in the aforementioned references.

For learning the gene regulatory networks, the first-order Markov tDBN learning algorithm was employed with the MDL score, allowing at most two parents from the past. However, as the number of observations was small, there was not enough evidence for MDL to include more than one parent. Figure 3 presents the resulting networks in compact form, to facilitate comparison to networks inferred by other authors. Table 2 examines the identified gene interactions against the ones reported in other publications, in the form of matching percentages. Guo et al. (2007) predicted non-stationary undirected networks, while Dondelinger et al. (2013) inferred non-stationary DBN.

The results in Table 2 suggest that tDBN learning algorithm performed reasonably well. The embryonic and larval networks contain a significant number of known interactions that are present in at least one of the corresponding reported networks. On the other hand, the pupal and adult networks did not achieve this result. Some of the pupal interactions could be disjointly found on the networks of both authors, resulting in a higher combined matching rate. The adult network, however, retrieved few known interactions, even when comparing to both sources combined.

Table 2: Comparative structure learning results on *Drosophila melanogaster* data.

| Morphogenic stage                  | Embryonic | Larval | Pupal | Adult |
|------------------------------------|-----------|--------|-------|-------|
| Observed time-slices               | 31        | 10     | 18    | 8     |
| Matches (Guo et al., 2007)         | 25%       | 50%    | 40%   | 30%   |
| Matches (Dondelinger et al., 2013) | 75%       | 60%    | 30%   | 20%   |
| Matches (both sources)             | 75%       | 70%    | 60%   | 40%   |

As acknowledged by Dondelinger et al. (2013), an objective assessment regarding the accuracy of the learnt networks is not possible due to limited biological knowledge available, which leads to the absence of a gold standard. Furthermore, there are three reasons for which the obtained results should be interpreted carefully. First, despite the best efforts to follow the procedure in Zhao et al. (2006), the resulting preprocessed dataset was possibly not the same. Second, learnt interactions are suggestions of causal regulatory effects. Additional biological experiments are necessary for validating the inferred networks, as noted in Guo et al. (2007). Third, the small number of observations leads to the existence of many equivalent networks

with maximum score, but only one is reported by the tDBN learning algorithm.

## 5 CONCLUSION

We have presented a simple yet effective algorithm for learning the structure of DBN, jointly recovering the inter and intra time-slice connectivity. The tDBN learning algorithm has polynomial time complexity with respect to the number of attributes and can be applied to non-stationary Markov processes.

The stationary version of the algorithm achieved very good results on simulated datasets, showing to be competitive with state of the art algorithms in recovering underlying structures. Furthermore, encouraging results were obtained on real data with the non-stationary and higher-order Markov versions of tDBN, indicating a broad scope of applications for the proposed algorithm.

Finally, the application of our method to learn non-stationary processes could be improved in conjunction with change-point techniques, as investigated in Robinson and Hartemink (2010) and Dondelinger et al. (2010). In its current form, the tDBN learning algorithm cannot identify changes in the underlying distribution of data and thus the number of transition networks to learn, as well as adequate training data for each network, need to be previously specified.

## Acknowledgments

This work was supported by national funds through FCT, Fundação para a Ciência e a Tecnologia, under contracts LAETA (UID/EMS/50022/2013) and IT (UID/EEA/50008/2013), and by projects CancerSys (EXPL/EMS-SIS/1954/2013) and IntelGen (PTDC/DTP-FTO/1747/2012). SV acknowledges support by Program Investigador FCT (IF/00653/2012) from FCT, co-funded by the European Social Fund (ESF) through the Operational Program Human Potential (POPH).

## References

M. N. Arbeitman, E. E. M. Furlong, F. Imam, E. Johnson, B. H. Null, B. S. Baker, M. A. Krasnow, M. P. Scott, R. W. Davis, and K. P. White. Gene expression during the life cycle of *Drosophila melanogaster*. *Science*, 297 (5590):2270–2275, 2002.

J. A. Bilmes. Dynamic Bayesian multinets. In *Proc. of the 16th Conference on Uncertainty in Artificial Intelligence*, pages 38–45, 2000.

D. Chickering, D. Geiger, and D. Heckerman. Learning Bayesian networks: Search methods and experimental

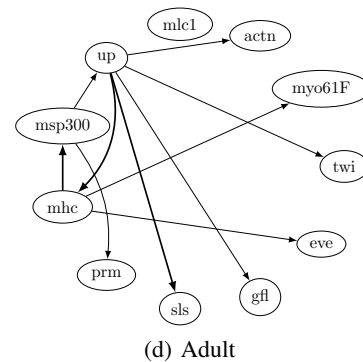
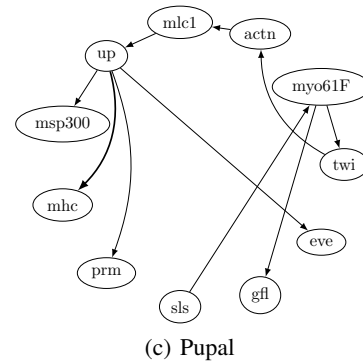
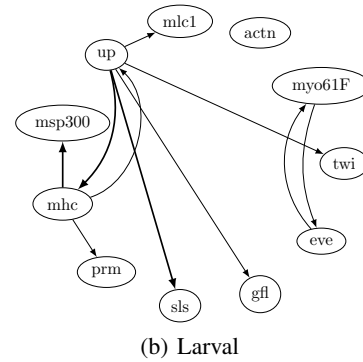
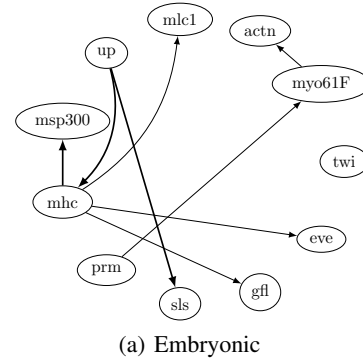


Figure 3: *Drosophila* gene regulatory networks identified by the tDBN learning algorithm. Networks are shown in compact form, where each edge represents a dependence between a node at time-slice  $t + 1$  and its parent at the previous time-slice  $t$ . Highlighted edges indicate a relation found in the majority of the morphogenic stages. Intra-slice edges are omitted.



- results. In *Proc. of the 5th International Workshop on Artificial Intelligence and Statistics*, pages 112–128, 1995.
- C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467, 1968.
- N. Dojer. Learning Bayesian networks does not have to be NP-hard. In *Mathematical Foundations of Computer Science 2006*, pages 305–314, 2006.
- N. Dojer, P. Bednarz, A. Podsiadło, and B. Wilczyński. BNFinder2: Faster Bayesian network learning and Bayesian classification. *Bioinformatics*, page btt323, 2013.
- F. Dondelinger, S. Lèbre, and D. Husmeier. Heterogeneous continuous dynamic Bayesian networks with flexible structure and inter-time segment information sharing. In *Proc. of the 27th International Conference on Machine Learning*, pages 303–310, 2010.
- F. Dondelinger, S. Lèbre, and D. Husmeier. Non-homogeneous dynamic Bayesian networks with Bayesian regularization for inferring gene regulatory networks with gradually time-varying structure. *Machine learning*, 90(2):191–230, 2013.
- J. Edmonds. Optimum branchings. *Journal of Research of the National Bureau of Standards B*, 71(4):233–240, 1967.
- A. Hartemink et al. Banjo: Bayesian network inference with Java objects. <http://www.cs.duke.edu/amink/software/banjo/>, 2005.
- N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine learning*, 29(2-3):131–163, 1997.
- N. Friedman, K. Murphy, and S. Russell. Learning the structure of dynamic probabilistic networks. In *Proc. of the 14th Conference on Uncertainty in Artificial Intelligence*, pages 139–147, 1998.
- M. Grzegorzcyk and D. Husmeier. Bayesian regularization of non-homogeneous dynamic Bayesian networks by globally coupling interaction parameters. In *Journal of Machine Learning Research Workshop and Conference Proceedings*, volume 22, pages 467–476, 2012.
- F. Guo, S. Hanneke, W. Fu, and E. P. Xing. Recovering temporally rewiring networks: A model-based approach. In *Proc. of the 24th international conference on Machine learning*, pages 321–328, 2007.
- D. Heckerman, D. Geiger, and D. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine learning*, 20(3):197–243, 1995.
- M. Kolar, L. Song, A. Ahmed, and E. P. Xing. Estimating time-varying networks. *The Annals of Applied Statistics*, 4(1):94–123, 2010.
- K. Murphy. The Bayes net toolbox for Matlab. *Computing Science and Statistics*, 33:2001, 2001.
- K. Murphy. *Dynamic Bayesian networks: representation, inference and learning*. PhD thesis, University of California, 2002.
- J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Representation and Reasoning Series. Morgan Kaufmann, 1988. ISBN 9781558604797.
- J. W. Robinson and A. J. Hartemink. Learning non-stationary dynamic Bayesian networks. *Journal of Machine Learning Research*, 11:3647–3680, 2010.
- R. Tarjan. Finding optimum branchings. *Networks*, 7(1): 25–35, 1977.
- N. Vinh, M. Chetty, R. Coppel, and P. Wangikar. GlobalMIT: Learning globally optimal dynamic Bayesian network with the mutual information test criterion. *Bioinformatics*, 27(19):2765–2766, 2011a.
- N. Vinh, M. Chetty, R. Coppel, and P. Wangikar. Polynomial time algorithm for learning globally optimal dynamic Bayesian network. In *Neural Information Processing*, pages 719–729, 2011b.
- W. Zhao, E. Serpedin, and E. R. Dougherty. Inferring gene regulatory networks from time series data using the minimum description length principle. *Bioinformatics*, 22(17):2129–2135, 2006.

---

# Learning and Inference in Tractable Probabilistic Knowledge Bases

---

**Mathias Niepert** and **Pedro Domingos**  
Department of Computer Science and Engineering  
University of Washington  
Seattle, WA 98195-2350, USA

## Abstract

Building efficient large-scale knowledge bases (KBs) is a longstanding goal of AI. KBs need to be first-order to be sufficiently expressive, and probabilistic to handle uncertainty, but these lead to intractable inference. Recently, tractable Markov logic (TML) was proposed as a non-trivial tractable first-order probabilistic representation. This paper describes the first inference and learning algorithms for TML, and its application to real-world problems. Inference takes time per query sublinear in the size of the KB, and supports very large KBs via parallelization and a disk-based implementation using a relational database engine. Query answering is fast enough for interactive and real-time use. We show that, despite the data being non-i.i.d. in general, maximum likelihood parameters for TML knowledge bases can be computed in closed form. We use our algorithms to build a very large tractable probabilistic KB from numerous heterogeneous data sets. The KB includes millions of objects and billions of parameters. Our experiments show that the learned KB outperforms existing specialized approaches on challenging tasks in information extraction and integration.

## INTRODUCTION

A knowledge base that continuously acquires knowledge and answers complex queries is a vision as old as the field of AI itself. The formal representation employed by such a KB needs to be at the level of first-order logic, in which inference is intractable. Many tractable subsets of first-order logic have been proposed (e.g., description logics, Horn KBs), but building large KBs further requires that the representation be probabilistic, for at least two reasons: most of the knowledge will necessarily be acquired from text, the Web, etc., which are inherently noisy and ambiguous, and

maintaining the consistency of large KBs is extremely difficult. First-order probabilistic representations like Markov logic have been developed, but they are again intractable, as are subsets like probabilistic Horn KBs and description logics. Recently, Domingos and Webb [2011] developed tractable Markov logic (TML), a subset of Markov logic that has expressiveness comparable to probabilistic Horn KBs and description logics but remains tractable. However, to date this was only a theoretical proposal, with no inference and learning algorithms beyond the basic inference required to prove tractability, and no real-world applications.

This paper develops highly scalable inference and learning algorithms for tractable probabilistic knowledge bases (TPKBs), using TML as the representation. After an initialization phase that computes the partition function of the TPKB in time linear in its size, queries are answered in sublinear time using a relational database implementation that allows for large disk-resident KBs and is easily parallelized. Maximum-likelihood parameter learning is done in closed form, and takes time linear in the size of the training data, which can therefore also be very large and disk-resident. Together these inference and learning algorithms make it possible to learn and reason with very large first-order probabilistic KBs.

As a test of our architecture and algorithms, we learn the structure and parameters of a TPKB from a multitude of very large data sets, and use it to answer a variety of queries. The resulting TPKB is larger than previous statistical relational models by at least an order of magnitude, with millions of objects and billions of parameters, yet allows for subsecond query answering times. Extensive experiments show the efficiency and effectiveness of the proposed framework for entity linking and resolution.

## RELATED WORK

Probabilistic description logic programs [17] combine DL programs under the answer set and well-founded semantics with independent choice logic [26]. Particular variants of light-weight description logics with probabilistic seman-

tics [11, 22, 24] have been proposed. However, these formalisms are too expressive to be tractable for the types of queries needed in large-scale applications, do not allow the modeling of data properties, and do not address the problem of parameter and structure learning.

TPKBs are related to **PROBLOG** [28] a probabilistic logic programming language that is generally intractable. **URDF** [19] is based on weighted **MAX-SAT** algorithms and is in principle intractable. It also does not support queries that ask for a distribution over objects, classes and relations given a subset of their attributes. Infinite relational models (IRMs) are non-parametric hierarchical models but are not tractable in general [13]. We leverage the hierarchy of TPKBs to estimate parameters on the level of classes and relation more robustly. This is related to shrinkage [18].

TPKBs are different from probabilistic databases [31] in that they represent general knowledge attached to class and relation hierarchies and do not assume tuple independence, that is, the independence of facts in the knowledge base. Moreover, inference in a TPKB is always at most linear in the size of the TPKB whereas inference complexity in PDBs depends on the query expression and can be costly.

Open information extraction [7] and other IE projects [2, 15] often use ad-hoc approaches and heuristics and do not provide a consistent joint distribution and query language. There exist several statistical relational systems employing relational database technology to facilitate queries over structured data [23, 33, 25]. However, the proposed systems are intractable. Recent work on statistical relational learning has focused on tractable probabilistic graphical models, that is, probabilistic models for which inference is efficient by design. Examples are relational sum-product networks [20], particular tractable fragments of probabilistic logics [32], and probabilistic soft logic [14]. None of these languages feature disk-based, sublinear inference algorithms or has been used to build a real-world tractable probabilistic knowledge base. There is related work in the context of information extraction and relation learning. Notable representative publications of this line of research are tensor factorizations of **YAGO** [21] and universal schemas [29]. These approaches do not facilitate a consistent probabilistic semantics and expressive query languages.

## TRACTABLE PROBABILISTIC KNOWLEDGE BASES

Tractable Markov logic (TML) [4] is a subset of Markov logic exploiting probabilistic class and part hierarchies to control complexity. In TML the domain is decomposed into parts, each part is drawn probabilistically from a class hierarchy, the part is further decomposed into parts according to its class, and so on. More recently, TML was extended

so as to also handle existence uncertainty [34]. Tractable probabilistic knowledge bases (TPKBs) accomplish this by defining a possible world as a set of objects and the truth values of the atoms in which they occur as arguments, rather than just as a set of truth values over a fixed universe of objects. TPKBs form the basis for the new algorithms in this paper. We extend them with attributes that can have discrete and continuous distributions. Attributes allow us to model typical properties of objects in real-world knowledge bases such as images, geographical locations, and mentions.

### Syntax

A TPKB is a set of class and object declarations. A class declaration specifies the subclasses, parts, attributes, and relations of the class, along with their distributions. It has the following form:

```

Class C {
  subclasses  $S_1 w_1, \dots, S_n w_n$ ;
  parts  $P_1[c_1] C_1, \dots, P_\ell[c_\ell] C_\ell$ ;
  attributes  $A_1 D_1 u_1, \dots, A_m D_m u_m$ ;
  relations  $R_1(\dots) v_1, \dots, R_k(\dots) v_k$ ;
}

```

Subclass declarations specify the direct subclasses (children) of  $C$  and their real-valued weights. Direct subclasses are assumed to be mutually disjoint and form a partition of the superclass.

Part declarations specify the parts every instance of  $C$  must have. Part declarations consist of a part name  $P_i$ , the part's class  $C_i$ , and the number  $c_i$  of parts of that type where  $c_i$  is optional and 1 by default. Two classes such that one is a descendant of the other in the class hierarchy never have a part with the same name.

Every attribute  $A$  has a domain  $D_i$  (e.g.,  $\{0, \dots, 10\}, \mathbb{R}$ ) and a weight function  $u_i : D_i \rightarrow \mathbb{R}$ . The weight function for a continuous domain must be efficiently integrable.

Each relation  $R_i(\dots)$  has the form  $R_i(P_a, \dots, P_z)$  where each of  $P_a, \dots, P_z$  is a part of  $C$ , that is, one of  $P_1, \dots, P_\ell$ , and the  $v_i$ 's are real-valued weights. Weights determine the probability that an object belongs to a direct subclass or that a relation is true as specified below. If a relation weight does not appear, then the relation is hard, that is, it must hold in every possible world.

Objects of the TPKB are introduced as parts, subparts, sub-subparts, etc., of a single *top object*. The top object is the sole instance of its class which is named the *top class*. The top class does not have superclasses. Given a set of class declarations, we can define the following concepts.

The *class hierarchy* is a directed graph whose nodes correspond to classes and whose edges correspond to direct

subclass relationships. The class hierarchy must be a forest.

The *part decomposition* is a directed graph whose nodes correspond to parts and with edges from each object to each of its possible parts. The part decomposition must be a tree with the top object as root node.

In addition to class declarations, TPKBs have object declarations which introduce evidence by specifying the names of the object as well as its subclass memberships, attribute values, and relations. An object declaration has the form

```
Path Name {
  S1, ..., Sn;
  A1 = D1, ...;
  R1(...), ..., ¬R1(...), ...;
}
```

where Path is the object's path from the top object in the part decomposition and Name the name given to the object. For instance, for a TPKB with top object World, World.Country<sub>1</sub>.State<sub>5</sub>.Capital Olympia { ... } is the declaration for object World.Country<sub>1</sub>.State<sub>5</sub>.Capital giving it the name Olympia. If, for instance, Country<sub>1</sub> was previously declared to be World.USA and State<sub>5</sub> to be USA.Washington, then Path could also be Washington.Capital.

The statement S<sub>1</sub>, ..., S<sub>n</sub>; expresses that S<sub>1</sub>, ..., S<sub>n</sub> are the only classes the object can be an instance of. Each statement A<sub>i</sub> = D, with D ∈ D<sub>i</sub>, is an attribute fact and denotes that the attribute is known to have value D. For every attribute A<sub>i</sub> of the object there exists at most one fact A<sub>a<sub>i</sub></sub> = D. Each statement R<sub>i</sub>(...) is a relation fact and denotes that the relation is known to be true, and each statement ¬R<sub>i</sub>(...) is a relation fact that denotes that the relation is known to be false. Object declarations are optional and may be empty, that is, they may have the form Path Name { }. Empty object declarations introduce evidence by declaring the object to exist in all possible worlds. Object declarations have the purpose of naming objects and introducing evidence and must be compatible with the class declarations. Table 1 depicts a typical set of class and object declarations.

## Semantics


The possible objects of a TPKB are the top object and every other object in the part decomposition. The Herbrand universe of a TPKB is the set of constants representing classes, paths representing objects, and constants representing attribute values. An *n*-ary *predicate* grounded with elements of the Herbrand universe is an *atom*. We define the binary predicate Is, where atom Is(O, C) is true if object O is of class C and false otherwise. Likewise, A(O, D) is an atom which is true if attribute A has value D for object O. Finally,


```
Class Country {
  parts Capital City, President Person;
  attributes area ℝ u1, image Images u2;
}

Class Person {
  subclasses Politician 0.6, ..., Actor 1.1;
  attributes age ℕ u3, image Images u4;
}

Class City {
  parts Mayor Person, PoliceChief Person;
  attributes area ℝ u5, image Images u6;
  relations reportsTo(PoliceChief,Mayor) 2.3;
}

World.Country1 USA {
  area = 9857306;
}

USA.Capital WashingtonDC {
  area = 77, image=;
}

WashingtonDC.Mayor MurielBowser {
  age = 42, image=;
}

WashingtonDC.PoliceChief CathyLanier { }
```

Table 1: Example class and object declarations of a TPKB. area is an attribute that models the area of an entity in square kilometers, and images a collection of images depicting entities.

R(O, ...) is an atom which is true if the relation R(...) is true for object O and false otherwise. We refer to class membership, attribute, and relation atoms and their negations as literals. A TPKB is a DAG of objects and their properties (classes, attributes, relations), and a possible world is a subtree of the DAG with values for the attributes and relations. More formally, a possible world **W** of a TPKB with top object O<sub>0</sub> and top class C<sub>0</sub> is a set of literals such that:

1. Is(O<sub>0</sub>, C<sub>0</sub>) ∈ **W**.
2. If Is(O, C) ∈ **W**, then
  - (a) if C has direct subclasses, then Is(O, S) ∈ **W** for exactly one direct subclass S and ¬Is(O, S') ∈ **W** for every other direct subclass S';
  - (b) for every part P of O declared to be of class C', Is(O.P, C') ∈ **W**;
  - (c) for every attribute A<sub>i</sub> declared for class C there is exactly one D ∈ D<sub>i</sub> with A<sub>i</sub>(O, D) ∈ **W** and ¬A<sub>i</sub>(O, D') ∈ **W** for every other D' ∈ D<sub>i</sub>; and
  - (d) for every relation R(...) declared for class C either R(O, ...) ∈ **W** or ¬R(O, ...) ∈ **W**; if R is hard, then R(O, ...) ∈ **W**.
3. No other literals are in **W**.

The set of objects that exist in a possible world is exactly the objects that occur as arguments of the class membership, relation, and attribute predicates. We write  $\mathcal{W}$  to denote the set of all possible worlds.

We write  $\text{Subs}(C)$ ,  $\text{Parts}(C)$ ,  $\text{Atts}(C)$  and  $\text{ReIs}(C)$  to denote the direct subclasses, parts, attributes, and relations declared for class  $C$ .

A possible subworld for object  $O$  and class  $C$  is defined as above with  $O_0$  replaced by  $O$  and  $C_0$  replaced  $C$ . The unnormalized distribution  $\phi$  over possible subworld  $\mathbf{W}$  without evidence, that is, without object declarations, is defined recursively as  $\phi(O, C, \mathbf{W}) = 0$  if  $\neg \text{Is}(O, C) \in \mathbf{W}$  or if a relation  $R$  of  $C$  is hard and  $\neg R(O, \dots) \in \mathbf{W}$  and otherwise as

$$\phi(O, C, \mathbf{W}) = \left( \sum_{S_i \in \text{Subs}(C)} e^{w_i} \phi(O, S_i, \mathbf{W}) \right) \times \left( \prod_{P_i \in \text{Parts}(C)} \phi(O, P_i, C_i, \mathbf{W})^{n_i} \right) \times \left( \prod_{A_i \in \text{Atts}(C)} \alpha(O, A_i, \mathbf{W}) \right) \times \left( \prod_{R_i \in \text{ReIs}(C)} \rho(O, R_i, \mathbf{W}) \right),$$

where  $\alpha(O, A_i, \mathbf{W}) = e^{u_i(D)}$  if  $A_i(O, D) \in \mathbf{W}$ . Moreover,  $\rho(O, R_i, \mathbf{W}) = e^{v_i}$  if  $R_i(O, \dots) \in \mathbf{W}$  and  $\rho(O, R_i, \mathbf{W}) = 1$  if  $\neg R_i(O, \dots) \in \mathbf{W}$ . If an object has no subclasses, parts, attributes, or relations, then the corresponding products in the above equation evaluate to 1.

The unnormalized probability of a possible world  $\mathbf{W}$  is now  $\phi(O_0, C_0, \mathbf{W})$ . The sub-partition function for object  $O$  and class  $C$  is

$$Z_{\mathcal{K}_{O,C}} = \sum_{\mathbf{W} \in \mathcal{W}} \phi(O, C, \mathbf{W}).$$

If  $O$  is the top object and  $C$  the top class, we simply write  $Z_{\mathcal{K}}$ . The probability of a possible world  $\mathbf{W}$  is

$$P(\mathbf{W}) = \frac{\phi(O_0, C_0, \mathbf{W})}{Z_{\mathcal{K}}}.$$

Sum-product networks (SPNs) are a class of deep probabilistic models in which one can perform fast exact inference [27]. An SPN is recursively constructed by forming sums and products of univariate distributions, and its partition function can be computed in time linear in its size. Every TPKB maps into a corresponding SPN such that the distribution of the TPKB is equivalent to the distribution of the SPN. The construction follows the recursive definition of the distribution over possible worlds  $\phi(O_0, C_0, \cdot)$ . Figure 1 depicts the SPN corresponding to the example TPKB

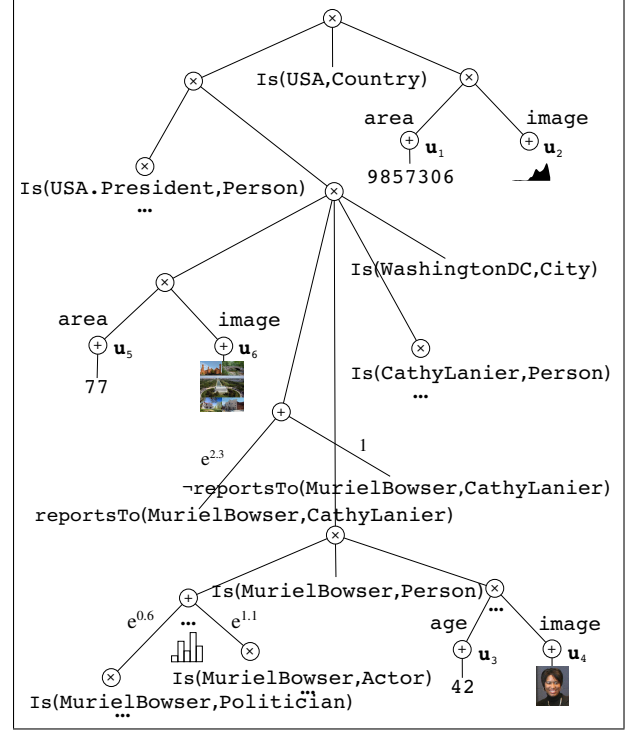


Figure 1: The SPN corresponding to the TPKB in Table 1.

of Table 1. The *size* of a TPKB is the number of objects times the number of classes each object is possibly an instance of. The proof of the following theorem can be found in the appendix.

**Theorem 1.** *The partition function of a TPKB can be computed in time linear in its size.*

We can also show that TPKBs are at least as expressive as the class of SPNs.

**Theorem 2.** *For every SPN there exists a TPKB with the same distribution and the size of the TPKB is linear in the size of the SPN.*

Hence, TPKBs include a large number of high-treewidth graphical models. These high-treewidth distributions are common in the real world, occurring whenever two subclasses of the same class have different subparts. TPKBs are non-trivial relational models that capture relational dependencies, multiple objects, existence uncertainty, etc. Also, even though data is not i.i.d. in general, closed-form maximum likelihood learning is possible given the TPKB's structure and complete data. Lifting in TPKBs is straightforward and makes the complexity of inference independent of the number of evidence-free objects.

The key assumption that TPKBs make to ensure tractability is that an object's attributes and relations are independent given its class. As a result, some distributions cannot be represented compactly. For example, an Ising model with

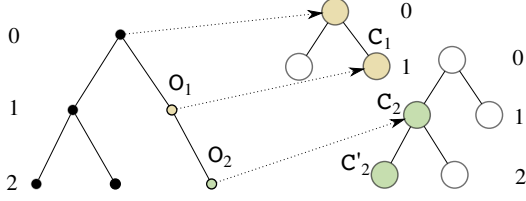


Figure 2: A part decomposition (left) with 6 objects and the class hierarchy (right) with two trees and 8 classes. Dashed arrows indicate which object was declared with which class. The numbers indicate the depth  $\ell$  of the layers in the part decomposition (left) and of the trees of the class hierarchy (right).

arbitrary structure may require an exponential number of classes to represent.

## INFERENCE

Object declarations introduce evidence in form of subclass, attribute, and relation facts and therefore influence the distribution of a TPKB by reducing the number of possible worlds with non-zero probability. The TPKB’s partition function is now the sum of the unnormalized probabilities of all possible worlds not contradicting the evidence. For a TPKB  $\mathcal{K}$  and a set of facts  $E$ , we write  $Z_{\mathcal{K}}(E)$  for the corresponding partition function.

There are several possible types of queries. For instance, we can ask for  $P(Q | E)$ , where  $E$  and  $Q$  are sets of facts pertaining to objects in  $\mathcal{K}$ . The answers to these queries can always be tractably computed as ratios of partition functions:

$$P(Q | E) = \frac{Z_{\mathcal{K}}(Q \cup E)}{Z_{\mathcal{K}}(E)}.$$

We can also perform MAP inference, that is, ask for the most probable possible world given evidence. This is possible in a TPKB simply by replacing sums over subclasses with maxes and performing a traceback.

For real-world knowledge bases with millions of objects and tens of thousand of relations and attributes, disk-based and parallel inference algorithms with sublinear running time are required. We now introduce a novel inference algorithm with these characteristics. To the best of our knowledge, this is the first sublinear exact inference algorithm that scales to millions of objects and is fully implemented using a relational database. The algorithm performs several relational queries, which allows us to take advantage of query evaluation plans, index structures, and caching strategies. Intuitively, the structure of the query expressions resembles the corresponding sum-product network where sum nodes correspond to join-and-sum operations and product nodes to join-and-multiply operations.

There are two distinct inference algorithms. The first one computes the partition function and, as a by-product, all sub-partition functions for objects and their possible classes (stored in table  $T_Z$ ) and the values of all sum nodes (stored in table  $T_{+S}$ ). It does this once offline, in time linear in the size of the TPKB.

The second algorithm computes, given evidence  $E$ , the *ratio* of the partition function with and without  $E$  (the marginal probability of  $E$ ), by propagating ratios of sub-partition functions with and without evidence.

To illustrate the algorithm, let us consider Figure 2 which depicts the part decomposition and the class hierarchy of a small TPKB. Every fact given by evidence  $E$  specifies either (a) the value of an attribute; (b) the truth value of a relation; (c) or the possible classes of an object. Evidence changes the sub-partition functions involving objects to which these attributes, relations, or subclasses apply. For instance, in Figure 2, if the sub-partition function for object  $O_2$  and its possible class  $C'_2$  is  $e^{w_s} e^{w_p} e^{w_a} e^{w_R}$ , and if evidence changes the factor pertaining to attributes from  $e^{w_a}$  to  $e^{w'_a}$  and leaves all other factors the same, then the ratio of the sub-partition functions with and without evidence is  $e^{w'_a} / e^{w_a}$ . The evidence also impacts the ratios of sub-partition functions with and without evidence,  $Z_{\mathcal{K}}^{(O_2, C)}(E) / Z_{\mathcal{K}}^{(O_2, C)}$ , for every superclass  $C$  of  $C'_2$  the object  $O_2$  can be an instance of (here: class  $C_2$ ). In Figure 2 these classes are depicted as shaded nodes in the second tree of the class hierarchy. Moreover, since  $O_2$  is a part of object  $O_1$ , the ratios of sub-partition functions for object  $O_1$  in layer 1 and the class it was declared with also change, and so on. Fortunately, we only need to compute the ratios for objects that are ancestors of the object whose sub-partition function is impacted by the evidence.

The inference algorithm propagates these ratios of sub-partition functions bottom-up through the TPKB’s structure to obtain  $Z_{\mathcal{K}}(E) / Z_{\mathcal{K}} = P(E)$ . The relational queries computing these ratios of sub-partition functions only involve tables changed by the evidence and, therefore, the complexity of the algorithm depends on the evidence and not the size of the TPKB.

The inference algorithm is shown in Algorithm 1. To avoid confusion with relations and attributes of the TPKB, we use the terms *table* for relations and *columns* for attributes in the relational algebra. Table  $T_Z(\text{Path}, \text{Class}, \text{subZ})$  with unique key  $(\text{Path}, \text{Class})$  represents the TPKB’s sub-partition functions. Hence,  $(O, C, Z) \in T_Z$  if and only if  $O$  can be an instance of class  $C$  and  $Z_{\mathcal{K}}(O, C) = Z$ . Analogously, we store the ratios of sub-partition function with and without evidence  $E$  in table  $T_Z^E$  with the same schema. We often write  $T_Z(O, C)$  to denote the value of the unique sub-partition function for object  $O$  and its possible class  $C$ .

Layer  $\ell$  of the part decomposition consists of all objects with depth (distance to the root node)  $\ell$ . Table

---

**Algorithm 1** Computes the probability of evidence  $E$  at each sub-partition function of objects in layer  $\ell$ .

---

```

1:  $T \leftarrow T_{\text{Is}}^{\ell, E}$ 
2:  $k \leftarrow$  maximum height of class hierarchy
3: if  $\ell <$  height of part decomposition then
4:    $T' \leftarrow T_{\text{Part}}^{\ell} \bowtie T_Z^E$ 
5:    $T_{\times P}^E \leftarrow \pi_{(\text{Path}, \text{Class}, \text{subZ})}(\text{G}_{\text{MUL}(\text{subZ})}(T'))$ 
6:    $T \leftarrow \otimes (T, T_{A_1}^E, \dots, T_{A_m}^E)$ 
7:    $T \leftarrow \otimes (T, T_{R_1}^E, \dots, T_{R_n}^E)$ 
8:    $T \leftarrow \otimes (T, T_{\times P}^E)$ 
9:    $T' \leftarrow \emptyset$ 
10: for each  $t \leftarrow (O, C, C', Z) \in T$  do
11:    $T_Z^E \leftarrow T_Z^E \cup \{(O, C, Z)\}$ 
12:   if  $C = C'$  then
13:      $T \leftarrow T - \{t\}$ 
14:   else if  $C$  is in layer  $k$  of the class hierarchy then
15:      $t \leftarrow (O, C, C', T_Z(O, C)(1 - Z))$ 
16:      $T' \leftarrow T' \cup \{t\}$ 
17:   if  $T \neq \emptyset$  then
18:      $T \leftarrow T - T'$ 
19:      $T' \leftarrow \oplus_{(\text{Path}, \text{Superclass})}(\otimes(T', T_{\text{Sub}}^{k-1}))$ 
20:      $T' \leftarrow \pi_{(\text{Path}, \text{Superclass} \rightarrow \text{Class}, \text{Class}', \text{subZ})}(T')$ 
21:     for each  $(O, C, C', Z) \in T'$  do
22:        $T \leftarrow T \cup (O, C, (T_{+S}(O, C) - Z)/T_{+S}(O, C))$ 
23:      $k \leftarrow k - 1$ 
24:   goto 6

```

---

$T_{\text{Is}}^{\ell, E}(\text{Path}, \text{Class}, \text{Class}')$  represents, for every object  $O$  in layer  $\ell$ , its possible leaf classes  $C$  in the class hierarchy, and the classes  $C'$  it is declared with. The table, however, only stores objects and classes whose sub-partition function is impacted by the evidence  $E$ . These sub-partition functions can be efficiently computed using the structure of the TPKB.

Table  $T_{\text{Part}}$  represents the part decomposition of the TPKB. This table is used in lines 4 and 5 of Algorithm 1 to compute the table  $T_{\times P}^E(\text{Path}, \text{Class}, \text{subZ})$  with  $(O, C, Z) \in T_{\times P}^E$  if and only if object  $O$  is in layer  $\ell$  of the part decomposition,  $O$  is declared as instance of class  $C$ , and  $Z$  is the product of the ratios of sub-partition functions with and without evidence of  $O$ 's parts according to  $C$ .

Lines 6, 7, and 8 evaluate the attribute, relation, and part product nodes of the TPKB's SPN (the three product nodes in Figure 1). The table  $T_A^E(\text{Path}, \text{Class}, \text{subZ})$  stores, for each object and its possible classes, the ratio of the exponentiated summed weights of the attribute  $A$  with and without evidence  $E$ . The tables  $T_{R_i}^E(\text{Path}, \text{Class}, \text{subZ})$  are the analogous tables for relations. The operator  $\otimes$  computes a left outer join on the columns  $\text{Path}$  and  $\text{Class}$ , and multiplies the values of the tables'  $\text{subZ}$  columns. The join queries of lines 5 and 6 only involve attribute and relation tables changed by the evidence.

Line 11 stores the computed ratios of sub-partition functions with and without evidence to table  $T_Z^E$ . In Line 12 it is tested, for each object  $O$ , whether we have reached the node of the class  $O$  was declared with. If this is the case, we remove the corresponding tuple from  $T$  since the computation of sub-partition functions that involve these objects is finished.

Lines 19 and 20 evaluate the SPN's sum nodes. First, we perform a left outer join of the tables  $T$  and  $T_{\text{Sub}}^k$  to multiply the children of each sum node with their respective weight. Table  $T_{\text{Sub}}(\text{Class}, \text{Superclass}, \text{weight})$  represents the class hierarchy and the subclass weights. For each  $\text{Superclass}$  in layer  $k$  of the class hierarchy,  $T_{\text{Sub}}^k(\text{Class}, \text{Superclass}, \text{weight})$  stores the weight for each of its subclasses. Second, the operator  $\oplus$  performs a grouping on the columns  $\text{Path}$  and  $\text{Superclass}$  and sums the values of column  $\text{subZ}$  for each of the groups. Line 20 renames the column  $\text{Superclass}$  to  $\text{Class}$  and projects out superfluous columns. The result of these operations are the values  $Z$  by which the sub-partition function at the sum nodes is reduced by the evidence. Line 22 computes the ratio of the sum node for object  $O$  and class  $C$  with and without evidence:  $(T_{+S}(O, C) - Z)/T_{+S}(O, C)$ .

Steps 6-22 are repeated until we have computed the ratio of the sub-partition function with and without evidence for every object in layer  $\ell$  and each of its possible classes.

After running Algorithm 1 for  $\ell = h, \dots, 0$ , where  $h$  is the height the part decomposition, we have that  $T_Z^E(O_0, C_0) = Z_{\mathcal{K}}(E)/Z_{\mathcal{K}} = P(E)$ .

**Theorem 3.** *Let  $\mathcal{K}$  be a TPKB and let  $n(E)$  be the number of sub-partition functions changed by a set of facts  $E$ . Algorithm 1 computes the probability  $P(E)$  in time  $O(n(E))$  and independent of the size of  $\mathcal{K}$ .*

*Proof sketch.* This can be shown with a nested proof by induction on (a) the height of the part decomposition and (b) the height of the class trees of each object.  $\square$

Hence, inference time depends on the number of sub-partition functions changed by the evidence. This approach is different from caching previously computed values [10]. While we exploit this idea as well, the major advancement is the propagation of only those ratios of partition functions that are changed by the evidence.

Algorithm 1 can be parallelized by leveraging the structure of the TPKB's part decomposition and class hierarchy. First, the executions of Algorithm 1 on distinct trees of the class hierarchy are independent and can be performed in parallel. Moreover, for each layer of the part decomposition, the execution of the algorithm for an object in the layer is independent of the execution for any other object in the same layer. More generally, for any two objects, Algorithm 1 can be executed independently until we reach a

layer with the parent of the two objects. By using relational database systems for query processing, we can take advantage of existing parallelization strategies that detect these types of decompositions. We show below that this reduces the query evaluation time on multi-core architectures.

## LEARNING

The TPKB’s parameters are the weights associated with attributes, relations, and subclasses. The training data is an i.i.d. sample of possible worlds. Hence, a sample  $\{\mathbf{W}_1, \dots, \mathbf{W}_N\}$  is a multiset of  $N$  i.i.d. possible worlds. Note that, even if the sample consists of only one possible world (a mega-example), we can often still obtain robust parameter estimates due to the structure of a TPKB which features a large number of objects with the same class and, therefore, the same attributes and relations.

We derive closed-form maximum likelihood (ML) estimators of the parameters. We choose ML parameters that lead to a partition function with value 1. Since for each possible world, the classes for all objects are known and every object can only be an instance of one of a set of sibling classes, the expression for the likelihood is a product of factors. The log-likelihood, therefore, is a sum of terms and differentiating it with respect to a particular parameter leaves only the terms involving one particular attribute, at which point the MLE can be found in closed form for exponential family distributions.

Let  $n(C)$  be the number of objects  $O$  in all possible worlds in the training data such that  $O$  is an instance of class  $C$ . For each class  $C$ , the probabilities of its subclasses  $S_1, \dots, S_n$  are governed by a categorical distribution. The standard ML estimates of  $P(S_i|C)$ , therefore, are  $n(S_i)/n(C)$ . Due to the parameterization of TPKBs, the MLEs are the logarithms of these estimates. Hence  $\text{MLE}(w_i) = \log(n(S_i)/n(C))$ .

Analogously, we can derive the ML estimates for attributes and relations. For categorical attributes, let  $n_A(C, D)$  be the number of objects  $O$  in all possible worlds in the training data such that  $O$  is an instance of class  $C$  and has value  $D$  for attribute  $A$ . Moreover, let  $n_A(C)$  be the number of objects  $O$  in all possible worlds such that  $O$  is an instance of class  $C$  and has *some* value for attribute  $A$ . If attribute  $A$  is declared for class  $C$  and for none of  $C$ ’s superclasses, then  $\text{MLE}(\mathbf{u}(D)) = \log(n_A(C, D)/n_A(C))$ . If attribute  $A$  is declared for class  $C$ , then it is also (implicitly) declared for each of its subclasses. Let  $S$  be such a subclass of  $C$ . Then

$$\text{MLE}(\mathbf{u}(D)) = \log \frac{n_A(S, D)/n_A(S)}{n_A(C, D)/n_A(C)}.$$

These are ML estimates precisely because we set the empirical probability, that is, the probability of an instance of class  $S$  having attribute value  $D$ , to the exponentiated sum of the weight estimates for  $S$  and all its superclasses. If the distribution is identical for a class and one of its subclasses,

```

Class Capital {
  parts s City, o Country;
  attributes distance  $\mathbb{R} \mathbf{u}_1$ ;
}
Class Country {
  subclasses USA 1.1, ..., Italy 0.4;
  attributes area  $\mathbb{R} \mathbf{u}_3$ , image Images  $\mathbf{u}_4$ ;
}
Class Paris {
  founded  $\mathbb{N} \mathbf{u}_8$ ;
}

```

Table 2: Example class and object declarations of the TPKB based on the DBPEDIA ontology.

the MLE is 0 and we do not have to explicitly represent the attribute parameters for the subclass. This is advantageous as it results in sparser TPKBs. Note also that this parameterization is useful as we may not know an object’s most specific class during inference time.

The ML estimates for attributes with continuous distributions are derived analogously. As before, the ML estimate is based on the sufficient statistics for the attribute at class  $C$  and its superclass for which the attribute is also declared. For instance, for Gaussian variables, we compute the mean and standard deviation, and the MLE for the weight function for  $S$  subclass of  $C$  is

$$\text{MLE}(\mathbf{u}(x)) = \log \frac{(x - \mu_S)^2 / \sigma_S^2}{(x - \mu_C)^2 / \sigma_C^2}.$$

Relations are similar to Boolean attributes except that for relations we represent the positive case only. The weight of a relation at a class is the log ratio of the positive and negative count differences.

In order to smooth distributions that are defined on multiple levels of the class hierarchy, we recursively average the estimate for each class with the estimate for the superclass, with the combination weights determined by a form of cross-validation [18]. There are more sophisticated variants of shrinkage and other forms of hierarchical smoothing, but we leave this to future work.

So far, we have assumed that the training data is complete, that is, that for every object its class, its attribute values, and the truth values of relations are given. If attribute values, classes, or relation values are missing in the training data, we can use the EM algorithm [3], alternating between inferring the distributions of the missing values and finding the MLEs.

Structure learning involves the learning of the part decomposition and the class hierarchy. While this is beyond the scope of this paper, possible directions include adapting LearnSPN [9] and LearnRSPN [20] to TPKBs.



## TPKBS FOR INFORMATION EXTRACTION

There are several information extraction projects such as NELL [2], DBPEDIA [16, 1], the Google Knowledge Vault [5], and REVERB [7]. These systems parse large amounts of text and semi-structured data sources. Typical extractions are of the form  $\langle m_s, m_p, m_o \rangle$  where  $m_s$  is a *mention* of the subject,  $m_p$  of the predicate, and  $m_o$  of the object of a sentence. For instance,  $\langle \text{Obama, graduated from, Columbia} \rangle$ . A common problem is the grounding of these extractions in a canonical KB such as DBPEDIA. It is a challenging problem due to the inherent ambiguities of extractions and the problem of representing context. For instance, in the above example extractions, the mention *Obama* might refer to numerous individuals; *Columbia* to the district, the university, or a number of other possible entities; and *graduated from* might express that someone graduated from high school, college, etc. Only in conjunction with background knowledge is it possible to infer that the extraction mentions Barack Obama, 44th President of the US, and his being a graduate of Columbia University in New York.

We propose to use TPKBs to address the problem of joint entity linking. The TPKB has three objects  $s$ ,  $p$ , and  $o$ , representing the latent canonical subjects, predicates, and objects of extractions. Every possible class of the object  $p$  represents a relation type. For instance, DBPEDIA's canonical relation type `alumni`, which expresses a person having graduated from a university, is such a class. The representation of relation types as classes of object pairs is advantageous due to the type-token distinction: in order to perform entity linking one has to model objects as classes, not constants, since one predicts the class membership of  $s$  and  $o$  from their attributes. Since objects are represented as classes, we cannot use TPKB relations and, instead, represent relation types as classes of object pairs.

The objects  $s$  and  $o$  are declared as parts of object  $p$ . The classes of  $s$  and  $o$  depend on the class in which they were declared. For instance, in class `alumni`,  $s$  is declared to be of class `Person` and  $o$  to be of class `University`. In class `capital`,  $s$  is declared to be of class `populatedPlace` and  $o$  of class `City`. Objects  $s$  and  $o$  are declared as parts of  $p$  only in classes representing relations without subrelations.

The class hierarchy consists of three trees, representing the class structure of objects  $o$ ,  $s$ , and  $p$ , respectively. The leaf classes of the class trees of  $s$  and  $o$  are classes that represent canonical entities such as `BarackObama`. The structure of this TPKB allows us to model the dependencies between attributes, classes, and relation types in a tractable and principled way. It also allows us to disambiguate extractions based on geographical, temporal, and other numerical attributes in a principled manner.

Once the parameters of the TPKB are estimated, we can perform entity linking. For instance, given the extraction  $\langle \text{Obama, graduated from, Columbia} \rangle$  we compute the probability of the mention *Obama* referring to the canonical entity `BarackObama` using the query  $P(Q | E)$  with  $Q = \{Is(s, BarackObama)\}$  and  $E = \{mention(s, Obama), mention(p, graduated from), mention(o, Columbia)\}$ , where *mention* is an attribute modeling mentions.

### Experiments

We derived the class trees for objects  $s$  and  $o$  directly from the DBPEDIA ontology [16, 1]. The class tree for object  $p$  is derived from the relations in DBPEDIA, that is, every relation type in DBPEDIA is represented with a class object that  $p$  can be an instance of. In addition to the classes and relation types taken from the DBPEDIA ontology, we created one leaf class for each canonical entity. For instance, the DBPEDIA entity `BarackObama` is a leaf in the class hierarchy with attributes `birthYear`, `mention`, etc. Table 2 depicts a small selection of class declarations.

For several attributes such as `birthYear`, `elevation`, `geocoordinates`, etc. we used data from DBPEDIA to learn the parameters. To learn the attribute distributions for leaf classes, that is, classes modeling entities, we assumed a uniform distribution if, for one entity, more than one value was given for a particular attribute. For instance, if `Arnold` has values 1947 and 1946 for attribute `birthYear` then, following the maximum-likelihood principle, we assume that both values have probability 0.5. For the other classes, we pooled attribute values of the instances of each class and used histograms to model these distributions. For the attribute `mention` modeling mentions we used the WIKIPREP tool [8] to compute the conditional distribution of a canonical entity given a mention. We also learned the parameters of attributes for classes representing relation types. For instance, we introduced the attribute `diffBirthYear` which models a distributions over the absolute value of birth year differences.

The number of parameters of the resulting TPKB exceeds 1 billion and we model more than 1 million objects. We performed inference by running Algorithm 1 using a MySQL database system. Each query was answered in less than one second. The reduction in running time for computing the partition function was 31% for 2 cores, 47% for 4 cores, 49% for 6 cores, and 50% for 8 cores. We evaluated the learned TPKB empirically on two important problem classes.

### Entity Linking

For the entity linking experiments we used an existing gold standard [6] for aligning NELL triples to DBPEDIA entities. For each NELL triple of the form  $\langle m_s, m_p, m_o \rangle$ , we

| NELL relation              | Precision@1 |             |             | Recall@1    |             |       |
|----------------------------|-------------|-------------|-------------|-------------|-------------|-------|
|                            | WL          | TPKB1       | TPKB2       | WL          | TPKB1       | TPKB2 |
| ActorStarredInMovie        | 0.81        | <b>0.85</b> | <b>0.92</b> | 0.82        | 0.82        | 0.31  |
| AgentcollaboratesWithAgent | 0.82        | 0.83        | <b>0.91</b> | 0.86        | <b>0.87</b> | 0.20  |
| AnimalIsTypeofAnimal       | 0.86        | 0.86        | <b>0.99</b> | 0.86        | 0.86        | 0.71  |
| AthleteLedSportsTeam       | 0.89        | <b>0.91</b> | <b>0.93</b> | 0.86        | <b>0.87</b> | 0.37  |
| BankBankInCountry          | 0.82        | <b>0.87</b> | <b>0.93</b> | 0.76        | 0.76        | 0.10  |
| CityLocatedInState         | 0.80        | <b>0.85</b> | <b>0.95</b> | 0.81        | 0.82        | 0.64  |
| BookWriter                 | 0.82        | 0.83        | <b>0.92</b> | 0.81        | <b>0.82</b> | 0.73  |
| CompanyAlsoKnownAs         | 0.71        | 0.71        | <b>1.00</b> | 0.58        | <b>0.61</b> | 0.49  |
| PersonLeadsOrganization    | 0.79        | 0.81        | <b>0.92</b> | <b>0.75</b> | 0.71        | 0.68  |
| TeamPlaysAgainstTeam       | 0.81        | 0.81        | <b>1.00</b> | 0.81        | <b>0.83</b> | 0.70  |
| WeaponMadeInCountry        | 0.88        | <b>0.91</b> | <b>1.00</b> | 0.88        | 0.89        | 0.65  |
| LakeInState                | 0.90        | <b>0.91</b> | <b>1.00</b> | 0.90        | 0.90        | 0.84  |

| System | Prec@1      | Rec@1       |
|--------|-------------|-------------|
| PARIS  | 91.9        | 73.8        |
| TPKB1  | 85.3        | <b>75.2</b> |
| TPKB2  | <b>92.1</b> | 74.0        |

Table 3: Results for entity resolution experiments (left) and entity linking experiments (right). Bold numbers indicate significance (paired t-test;  $p < 0.05$ ) compared to the baselines.

performed the following two queries. The query (TPKB1)

$$P(\{Is(s, x), Is(o, y)\} \mid \{mention(s, m_s), mention(o, m_o)\})$$

asks for the marginal probability of  $s$  being entity  $x$  and  $o$  being entity  $y$ , conditioned on the NELL triple’s mentions. The result is a list of substitutions for variables  $x$  and  $y$  and the corresponding marginal probabilities.

We proceeded to manually align the subject and object mention with their classes in the TPKB. For instance, for the triple  $\langle \text{Obama, graduated from, Columbia} \rangle$ , we aligned Obama to the class Politician and Columbia to the class University. We then performed the previous query except that we added the set  $\{Is(s, C_s), Is(o, C_o)\}$  to the evidence (TPKB2). This query retrieves the probabilities of  $s$  being entity  $x$  and  $o$  being entity  $y$ , conditioned on the mentions and their class memberships. The results are given in Table 3 (left) and compared with a baseline given in [6]. There are other possible baselines such as matrix factorization methods [21, 29]. However, it is non-trivial to use these methods for the entity linking problem. This is because we have only mentions of entities and no data that links these mentions to relations and attributes in the canonical KBs. Precision@ $k$  and Recall@ $k$  is computed by retrieving the  $k$  most probable answer tuples. The results of the TPKB outperform the WikiLink baseline [6] substantially and are as efficient to compute. TPKB2, however, should only be used if high precision results are required.

### Entity Resolution

Entity resolution is the problem of determining whether two entities in two knowledge bases are equivalent. To evaluate the TPKB for entity linking we repeated the experiment of linking YAGO [12] to DBPEDIA conducted to evaluate the PARIS matching system [30]. Both knowledge bases use Wikipedia identifiers for their objects which gives us a large set of gold standard pairs for evaluation purposes. We manually aligned a set of attributes (datatype

properties) and classes between YAGO and the learned TPKB. We sampled 100,000 objects in YAGO, retrieved the aligned attributes for each object (labels, numerical attributes, etc.) and ran, for each entity, the query above, where we condition on the given value of attribute mention (TPKB1); and all other attributes for which a manual alignment existed (TPKB2). Table 3 shows that TPKBs are able to accurately link entities and compare favorably with specialized algorithms.

The learned TPKB is both efficient and accurate, outperforming existing problem-specific approaches.

## CONCLUSION

We presented a novel inference algorithm for TPKBs that is disk-based, parallel, and sublinear. We also derived closed-form maximum likelihood estimates for TPKB parameters. We used these results to learn a large TPKB from multiple data sources and applied it to information extraction and integration problems. The TPKB outperformed existing algorithms in accuracy and efficiency.

Future work will be concerned with more sophisticated smoothing approaches, the comparison of different learning strategies, and the problem of structure learning. We also plan to apply TPKBs to a wide range of problems that benefit from tractable probabilistic knowledge representations.

An open-source implementation of TPKBs is available at [alchemy.cs.washington.edu/lite2](http://alchemy.cs.washington.edu/lite2).

## ACKNOWLEDGMENTS

This research was partly funded by ONR grants N00014-13-1-0720 and N00014-12-1-0312, and AFRL contract FA8750-13-2-0019. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ONR, AFRL, or the United States Government.

## References

- [1] C. Bizer, T. Heath, and T. Berners-Lee. Linked data - the story so far. *Int. J. Semantic Web Inf. Syst.*, 5(3):1–22, 2009.
- [2] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. H. Jr., and T. M. Mitchell. Toward an architecture for never-ending language learning. In *Proc. AAAI*, pages 1306–1313, 2010.
- [3] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.
- [4] P. Domingos and W. A. Webb. A tractable first-order probabilistic logic. In *Proc. AAAI*, pages 1902–1909, 2012.
- [5] X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmman, S. Sun, and W. Zhang. Knowledge Vault: A web-scale approach to probabilistic knowledge fusion. In *Proc. SIGKDD*, pages 601–610, 2014.
- [6] A. Dutta, C. Meilicke, M. Niepert, and S. P. Ponzetto. Integrating open and closed information extraction: Challenges and first steps. In *Proc. NLP-DBPEDIA@ISWC*, 2013.
- [7] O. Etzioni, A. Fader, J. Christensen, S. Soderland, and Mausam. Open information extraction: The second generation. In *Proc. IJCAI*, pages 3–10, 2011.
- [8] E. Gabrilovich and S. Markovitch. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In *Proc. IJCAI*, pages 1606–1611, 2007.
- [9] R. Gens and P. Domingos. Learning the structure of sum-product networks. In *Proc. ICML*, pages 873–880, 2013.
- [10] V. Gogate and P. Domingos. Probabilistic theorem proving. In *Proc. UAI*, pages 256–265, 2011.
- [11] V. Gutiérrez-Basulto, J. C. Jung, C. Lutz, and L. Schröder. A closer look at the probabilistic description logic prob-el. In *Proc. AAAI*, pages 197–202, 2011.
- [12] J. Hoffart, F. M. Suchanek, K. Berberich, and G. Weikum. YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia. *Artif. Intell.*, 194:28–61, 2013.
- [13] C. Kemp, J. B. Tenenbaum, T. L. Griffiths, T. Yamada, and N. Ueda. Learning systems of concepts with an infinite relational model. In *Proc. AAAI*, pages 381–388, 2006.
- [14] A. Kimmig, S. H. Bach, M. Broecheler, B. Huang, and L. Getoor. A short introduction to probabilistic soft logic. In *Proc. NIPS Workshop on Probabilistic Programming: Foundations and Applications*, 2012.
- [15] J. Lehmann, D. Gerber, M. Morsey, and A.-C. N. Ngomo. Defacto - deep fact validation. In *Proc. ISWC*, pages 312–327, 2012.
- [16] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer. DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal*, 2014.
- [17] T. Lukasiewicz. Probabilistic description logic programs. *Int. J. Appr. Reas.*, 45, 2007.
- [18] A. McCallum, R. Rosenfeld, T. M. Mitchell, and A. Y. Ng. Improving text classification by shrinkage in a hierarchy of classes. In *Proc. ICML*, pages 359–367, 1998.
- [19] N. Nakashole, M. Sozio, F. M. Suchanek, and M. Theobald. Query-time reasoning in uncertain RDF knowledge bases with soft and hard rules. pages 15–20, 2012.
- [20] A. Nath and P. Domingos. Learning the structure of relational sum-product networks. In *Proc. AAAI*, pages 873–880, 2013.
- [21] M. Nickel, V. Tresp, and H.-P. Kriegel. Factorizing YAGO: Scalable machine learning for linked data. In *Proc. WWW*, pages 271–280, 2012.
- [22] M. Niepert, J. Noessner, and H. Stuckenschmidt. Log-linear description logics. In *Proc. IJCAI*, pages 2153–2158, 2011.
- [23] F. Niu, C. Ré, A. Doan, and J. W. Shavlik. Tuffy: Scaling up statistical inference in Markov logic networks using an RDBMS. *PVLDB*, 4(6):373–384, 2011.
- [24] J. Noessner and M. Niepert. Elog: A probabilistic reasoner for OWL EL. In *Proc. RR*, pages 281–286, 2011.
- [25] J. Noessner, M. Niepert, and H. Stuckenschmidt. Rockit: Exploiting parallelism and symmetry for map inference in statistical relational models. In *Proc. AAAI*, pages 739–745, 2013.
- [26] D. Poole. The independent choice logic and beyond. In *Probabilistic inductive logic programming*. Springer-Verlag, 2008.
- [27] H. Poon and P. Domingos. Sum-product networks: A new deep architecture. In *Proc. UAI*, pages 337–346, 2011.
- [28] L. D. Raedt, A. Kimmig, and H. Toivonen. Problog: a probabilistic prolog and its application in link discovery. In *Proc. IJCAI*, pages 2468–2473, 2007.
- [29] S. Riedel, L. Yao, B. M. Marlin, and A. McCallum. Relation extraction with matrix factorization and universal schemas. In *Proc. HLT-NAACL*, pages 74–84, 2013.
- [30] F. M. Suchanek, S. Abiteboul, and P. Senellart. PARIS: Probabilistic alignment of relations, instances, and schema. *PVLDB*, 5(3):157–168, 2011.
- [31] D. Suciu, D. Olteanu, R. Christopher, and C. Koch. *Probabilistic Databases*. Morgan & Claypool Publishers, 1st edition, 2011.
- [32] G. Van den Broeck. On the completeness of first-order knowledge compilation for lifted probabilistic inference. In *Proc. NIPS*, pages 1386–1394, 2011.
- [33] D. Z. Wang, M. J. Franklin, M. N. Garofalakis, and J. M. Hellerstein. Querying probabilistic information extraction. *PVLDB*, 3(1):1057–1067, 2010.
- [34] W. Webb and P. Domingos. Tractable probabilistic knowledge bases with existence uncertainty. In *Proc. AAAI Workshop on Statistical Relational AI*, 2013.

---

# Multi-Context Models for Reasoning under Partial Knowledge: Generative Process and Inference Grammar

---

**Ardavan S. Nobandegani**

Dept. of Electrical and Computer Engineering  
McGill University  
Montreal, QC H3A 0E9

**Ioannis N. Psaromiligkos**

Dept. of Electrical and Computer Engineering  
McGill University  
Montreal, QC H3A 0E9

## Abstract

Arriving at the complete probabilistic knowledge of a domain, i.e., learning how all variables interact, is indeed a demanding task. In reality, settings often arise for which an individual merely possesses partial knowledge of the domain, and yet, is expected to give adequate answers to a variety of posed queries. That is, although precise answers to some queries, in principle, cannot be achieved, a range of plausible answers is attainable for each query given the available partial knowledge. In this paper, we propose the Multi-Context Model (MCM), a new graphical model to represent the state of partial knowledge as to a domain. MCM is a middle ground between Probabilistic Logic, Bayesian Logic, and Probabilistic Graphical Models. For this model we discuss: (i) the dynamics of constructing a contradiction-free MCM, i.e., to form partial beliefs regarding a domain in a gradual and probabilistically consistent way, and (ii) how to perform inference, i.e., to evaluate a probability of interest involving some variables of the domain.

## 1 INTRODUCTION

At an abstract level, an individual (also referred to as a reasoner) is faced with a domain where by “domain” we simply mean a collection of propositions or concepts which are mathematically encoded as Random Variables (RVs). To arrive at the complete probabilistic knowledge of the domain, i.e., to learn how all RVs in the domain probabilistically interact with one another, is indeed a demanding task. In reality, an individual is often faced with a domain for which she merely possesses *partial* knowledge—that is, she only knows how *some* (not all) RVs in the domain interact. To make the setting under study more tangible, consider the following case. Suppose that the probabilistic knowledge of a domain is represented by a Probabilis-

tic Graphical Model (PGM)  $\mathcal{B}$ , e.g., a Bayesian Network (BN). Then the reasoner comes across a new RV, say  $\psi$ , and would like to incorporate it into  $\mathcal{B}$  so as to achieve the complete probabilistic knowledge of the new domain (which now also includes  $\psi$ ). However, incorporation of  $\psi$  into  $\mathcal{B}$  would require knowledge of how  $\psi$  is probabilistically related to all the RVs already present in  $\mathcal{B}$ ; a knowledge which may be, quite plausibly, unavailable to the reasoner. An interesting question that now arises is how to handle situations where only partial knowledge as to how  $\psi$  is probabilistically related to  $\mathcal{B}$  is available. An example would be when the reasoner merely knows how  $\psi$  interacts probabilistically with only one RV, say  $\phi$ , in  $\mathcal{B}$ .

In this paper, a graphical model, namely, the Multi-Context Model (MCM) is proposed to represent the setting in which only partial probabilistic knowledge of a domain is available to the reasoner. More specifically, MCM is a graphical language to represent settings in which the Joint Probability Distribution (JPD) over all RVs is not available, but what is available instead is the JPDs over a collection of subsets of RVs of the domain (referred to as sub-domains or *contexts*). These contexts are potentially overlapping, i.e., they could share some RVs. As pointed out elegantly in (Pearl 1990), “*this state of partial knowledge is more common, because we often begin thinking about a problem through isolated frames, paying no attention to interdependencies.*” Along the same line of thought, it is plausible to assume that the probabilistic knowledge of the domain at the early primitive stage consists of a collection of disjoint contexts and as the reasoner acquires more knowledge as to how the variables in the model are related to one another and thus probabilistically interact, contexts gradually go through a process very much like an evolution: contexts start to share some variables, overlaps begin to emerge and, once enough knowledge is obtained, a number of contexts could merge thereby giving rise to bigger contexts. This naturally raises the following fundamental question: How could a collection of consistent, probabilistically sound, and potentially overlapping contexts emerge *gradually* over the course of time? In an attempt to answer this question we present a generative process of constructing a contradiction-free

MCM. Finally, we would like to note that the special case where the whole domain is modeled as a single context corresponds to the conventional way of modeling the probabilistic knowledge of a domain using a single PGM, e.g., by some BN.

Another yet crucial question which we address in this work—which is another motivation behind the development of the MCM—is how the task of inference (i.e., the evaluation of some probability of interest which is hereafter referred to as *query*) should be carried out in a domain which is modeled according to some MCM. A query does not necessarily belong to any one of the contexts in particular and, in fact, may involve RVs from different contexts.

The paper is structured as follows. After introducing the notation in Sec. 2, we define in Sec. 3 the MCM and drawing on the notion of probabilistic conditioning, a generative process of constructing a contradiction-free MCM is discussed. Then, in Sec. 4 we elaborate on the problem of inference in a multi-context setting, i.e., in a domain whose probabilistic knowledge is encoded as an MCM. In Sec. 5 we discuss the relevant past work and comment on the proposed model. Finally, Sec. 6 concludes the paper.

## 2 TERMINOLOGY AND NOTATION

In this section we present the mathematical notation and the terminology employed in this paper. Random quantities are denoted by bold-faced letters; their realizations are denoted by the same letter but non-bold. More specifically, RVs are denoted by lower-case bold-faced letters, e.g.,  $\mathbf{x}$ , while random vectors are denoted by upper-case bold letters, e.g.,  $\mathbf{X}$ .  $Val(\cdot)$  denotes the set of values a random quantity can take, e.g.,  $Val(\mathbf{x})$  is the set of all possible realizations of the RV  $\mathbf{x}$ . In this paper, we assume that all random quantities are discrete.

The JPD over the RVs  $\mathbf{x}_1, \dots, \mathbf{x}_n$  is denoted by  $\mathbb{P}(\mathbf{x}_1, \dots, \mathbf{x}_n)$ ; when  $\mathbf{x}_1, \dots, \mathbf{x}_n$  comprise a vector  $\mathbf{X}$  then  $\mathbb{P}(\mathbf{X}) := \mathbb{P}(\mathbf{x}_1, \dots, \mathbf{x}_n)$ . We will use the notation  $\mathbf{x}_{1:n}$  to denote the sequence of  $n$  RVs  $\mathbf{x}_1, \dots, \mathbf{x}_n$ . To simplify presentation and to prevent our expressions from becoming cumbersome, we incur the following abuse of notation: We denote the probability  $\mathbb{P}(\mathbf{x} = x)$  by  $\mathbb{P}(x)$  for some RV  $\mathbf{x}$  and its realization  $x \in Val(\mathbf{x})$ . Also,  $\mathbb{P}(\bar{x}) := \mathbb{P}(\mathbf{x} \neq x) = 1 - \mathbb{P}(x)$  for some  $x \in Val(\mathbf{x})$ , i.e.,  $\mathbb{P}(\bar{x})$  is the probability that  $\mathbf{x}$  takes on any value other than  $x$ . For conditional probabilities we will use the notation  $\mathbb{P}(x|y)$  instead of  $\mathbb{P}(\mathbf{x} = x|\mathbf{y} = y)$ . Similar notations will be used for the case of random vectors, i.e.,  $\mathbb{P}(X) := \mathbb{P}(\mathbf{X} = X)$ ,  $\mathbb{P}(\bar{X}) := \mathbb{P}(\mathbf{X} \neq X) = 1 - \mathbb{P}(\mathbf{X} = X) = 1 - \mathbb{P}(X)$ , and  $\mathbb{P}(X|Y) := \mathbb{P}(\mathbf{X} = X|\mathbf{Y} = Y)$ .

The subscript  $\downarrow$  on a probability, e.g.,  $\mathbb{P}(x|y)_{\downarrow}$ , denotes the minimum value the probability can take subject to the constraints induced by the available probabilistic knowl-

edge. Likewise, the subscript  $\uparrow$  on a probability denotes the maximum value the probability can take. Finally, the operator  $[\cdot]^+$  gives the positive part of its argument, i.e.,  $[a]^+ := \max\{0, a\}$  for any real-valued  $a$ .

## 3 MULTI-CONTEXT MODEL

As explained earlier, a *domain* is simply the set of all Random Variables (RVs) at hand. A *context* comprises a collection of RVs for which their JPD is precisely known, see Fig. 1(a). In general, two contexts could be disjoint (Fig. 1(b)) or overlapping (Fig. 1(c)).

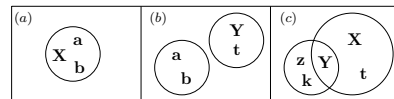


Figure 1: Graphical representation of contexts: (a) Context associated to  $\mathbb{P}(\mathbf{a}, \mathbf{b}, \mathbf{X})$ . (b) Two disjoint contexts associated to  $\mathbb{P}(\mathbf{a}, \mathbf{b})$  and  $\mathbb{P}(\mathbf{Y}, \mathbf{t})$ . (c) Two overlapping contexts associated to  $\mathbb{P}(\mathbf{X}, \mathbf{Y}, \mathbf{t})$  and  $\mathbb{P}(\mathbf{Y}, \mathbf{z}, \mathbf{k})$ . The random vector  $\mathbf{Y}$  is referred to as the *induced* part in Sec. 3.

A *Multi-Context Model (MCM)* encodes the probabilistic knowledge of a domain as a collection of possibly overlapping contexts. This enables the handling of situations in which comprehensive knowledge of a domain is not available, but partial information is, in the form of JPDs of some subsets of the domain. Let us first motivate the proposed MCM by entertaining a simple yet enlightening example.

### 3.1 MOTIVATING EXAMPLE

Consider a domain consisting of the RVs  $\mathbf{y}, \mathbf{z}$  in addition to a set of  $n$  RVs,  $\mathbf{x}_{1:n}$ . A reasoner has formed a partial belief as to the probabilistic connections between the variables of the domain. More specifically, the reasoner knows precisely the JPDs  $\mathbb{P}(\mathbf{y}, \mathbf{z})$  and  $\mathbb{P}(\mathbf{x}_{1:n})$  but not the JPD  $\mathbb{P}(\mathbf{y}, \mathbf{z}, \mathbf{x}_{1:n})$ . This setting is described by an MCM that consists of two disjoint contexts, one associated to RVs  $\mathbf{y}, \mathbf{z}$  and the other to  $\mathbf{x}_{1:n}$ , as shown in Fig. 2.

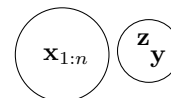


Figure 2: Problem statement as an MCM.

Assume that the following query is posed: Given the available information, what could be said about  $\mathbb{P}(y|x_i)$  for some  $i = 1, \dots, n$ ? The RVs  $\mathbf{y}$  and  $\mathbf{x}_i$  belong to different contexts, therefore, the JPD of  $\mathbf{y}$  and  $\mathbf{x}_i$ ,  $\mathbb{P}(\mathbf{x}_i, \mathbf{y})$ , is not available. The best one can hope for is to derive the range within which  $\mathbb{P}(y|x_i)$  varies, namely,  $[\mathbb{P}(y|x_i)_{\downarrow}, \mathbb{P}(y|x_i)_{\uparrow}]$ .

Let us for the moment assume the objective is to find  $\mathbb{P}(y|x_i)_\downarrow$ . Based on the conventional methodology, i.e., the approach adopted by past work (cf. (Andersen and Hooker 1990; 1994; Hansen et al. 1995) and references therein) one has to write down *all* the information as a list of linear equations and solve it as a Linear Program (LP). The main drawback of the conventional approach is that it cannot distinguish between what information is relevant and what is irrelevant for the posed query, and hence what needs to and what need not be considered in answering the query. The price for this is that the number of parameters required to merely formulate the query as an LP is exponential in  $n$ .

The key point, however, is that what information is relevant (or irrelevant) depends directly on the posed query, i.e., it is query-dependent. The main advantage of the proposed MCM over previous approaches is that it enables answering a query in a computationally efficient manner by distinguishing the relevant information from the irrelevant for the given query. This is realized through adopting the notion of *inference grammar*; a concept which will be systematically defined later. For our example, following the inference rule we will provide in Sec. 4.2, one can easily get  $\mathbb{P}(y|x_i)_\downarrow = \left[ \frac{\mathbb{P}(y) - \mathbb{P}(\bar{x}_i)}{\mathbb{P}(x_i)} \right]_+$ .

The task of inference in an MCM is carried out on two different levels, which makes the task more computationally efficient:

- (i) High-Level Reasoning: at this level, through the use of inference grammar, the relevant quantities are identified (e.g.,  $\mathbb{P}(y)$  and  $\mathbb{P}(\bar{x}_i)$  in the case of our example).
- (ii) Low-Level Reasoning: the relevant quantities, identified in (i), can then be computed by employing inference algorithms which take advantage of the potentially rich independence structure governing the contexts. For example, it could very well be the case that for the JPD associated to  $\mathbf{x}_{1:n}$  a large number of conditional independence relations hold. In that case, stating the derivation of  $\mathbb{P}(\bar{x}_i)$  (i.e.,  $1 - \mathbb{P}(x_i)$ ) as an LP would be computationally inefficient<sup>1</sup> but unnecessary. Indeed, the task of finding  $\mathbb{P}(\bar{x}_i)$  could be accomplished in a computationally efficient way using one of the many inference methods developed for probabilistic graphical models; a key point that the previous approaches do not take advantage of.

As a final step, in order to derive the lower/upper bound to the posed query, the quantities identified in (i) and subsequently calculated in (ii) are stated and solved as an LP.

The idea behind “high-level reasoning” will be explained and clarified further in Sec. 4.2 and 4.3, while the concept

<sup>1</sup>The number of parameters required just to state the problem as an LP is exponential in  $n$ .

of “low-level reasoning” will be discussed in Sec. 4.1.

### 3.2 GENERATIVE PROCESS OF CONTRADICTION-FREE MCMS

The objective of the generative process we describe in this section is to provide a way to consistently<sup>2</sup> construct contexts, in a sequential manner, over a set of RVs. The act of constructing a context, i.e., of assigning a JPD to a subset of RVs, corresponds to forming a *subjective*<sup>3</sup> belief over those RVs. In this light, the act of constructing multiple contexts corresponds to *gradually* forming subjective beliefs over a number of subsets of variables in the domain; hence every context symbolizes an established belief over the RVs involved in that context.

We introduce this problem by considering a simple case shown in Fig. 3(a). Suppose there are three RVs, namely,

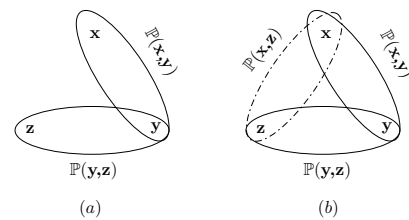


Figure 3: Generative process for contradiction-free Multi-Context Model. The dash-dotted contexts cannot be freely assigned.

$\mathbf{x}$ ,  $\mathbf{y}$ , and  $\mathbf{z}$ , present in the domain and let us consider the following question: Could one assign  $\mathbb{P}(\mathbf{x}, \mathbf{y})$  and  $\mathbb{P}(\mathbf{y}, \mathbf{z})$ , *freely* and *gradually* in a consistent manner, over the three variables without introducing any sort of contradiction? It is easy to verify that the answer is positive. Indeed, one could start off by assigning  $\mathbb{P}(\mathbf{x}, \mathbf{y})$ . This assignment would, of course, induce the marginal  $\mathbb{P}(\mathbf{y})$  and one can write  $\mathbb{P}(\mathbf{y}, \mathbf{z}) = \mathbb{P}(\mathbf{y})\mathbb{P}(\mathbf{z}|\mathbf{y})$ . Then, to complete this task, one would just need to proceed with assigning  $\mathbb{P}(\mathbf{z}|\mathbf{y})$ . This process could be referred to as a *generative* process of the assignment of  $\mathbb{P}(\mathbf{x}, \mathbf{y})$  and  $\mathbb{P}(\mathbf{y}, \mathbf{z})$  over  $\mathbf{x}$ ,  $\mathbf{y}$ , and  $\mathbf{z}$  without introducing any inconsistencies, in a gradual manner. Indeed, free-assignment refers to the act of freely assigning the non-induced, e.g.,  $P(\mathbf{z}|\mathbf{y})$ , part of the *to-be-formed* belief, e.g.,  $P(\mathbf{y}, \mathbf{z})$ . In other words, free-assignment signifies the observation that the already-formed belief does not impose any constraints on the non-induced part of the to-be-formed belief.

<sup>2</sup>That is, without introducing any form of contradictory result with respect to any probability assignment.

<sup>3</sup>One must not interpret the subjectivity of belief as “total disconnectivity from the reality.” Thus, we adopt the Bayesian interpretation of probability in this section. The avid reader is referred to (Chalmers 1976). An adherent to the frequentist interpretation of probability could think of contexts as being empirically constructed from a collection of data and thus skip Sec. 3.2 and proceed directly to the next section.

Let us now consider the case shown in Fig. 3(b). Could one assign  $\mathbb{P}(\mathbf{x}, \mathbf{y})$ ,  $\mathbb{P}(\mathbf{y}, \mathbf{z})$ , and  $\mathbb{P}(\mathbf{x}, \mathbf{z})$  freely and gradually in a consistent manner over the three variables without introducing any sort of contradiction? After some investigation, one can see that the answer is negative (Pearl 1985). Not surprisingly, the reason for this has to do with the existence of a loop in the model: once  $\mathbb{P}(\mathbf{x}, \mathbf{y})$  and  $\mathbb{P}(\mathbf{y}, \mathbf{z}) = \mathbb{P}(\mathbf{y})\mathbb{P}(\mathbf{z}|\mathbf{y})$  are assigned<sup>4</sup>, then  $\mathbb{P}(\mathbf{x}, \mathbf{z})$  cannot be assigned freely. This is due to the fact that  $\mathbb{P}(\mathbf{x}, \mathbf{z})$  has to satisfy some non-trivial conditions imposed by the already assigned contexts  $\mathbb{P}(\mathbf{x}, \mathbf{y})$  and  $\mathbb{P}(\mathbf{y}, \mathbf{z})$  (Pearl 1985).

In summary, whenever it comes to generating a new context, the JPD associated to that context has to be separated into two parts: (i) the part induced by the already existing contexts, and (ii) the part containing new variables which have never been so far associated to any context (i.e., non-induced part). The key point in the generation of contradiction-free MCMs is that the former part has to be induced by some context which, itself, is already present in the domain. That is, all the induced parts have to be already contained within some context. Otherwise, to include the induced parts—each constrained by the context it is already in—in a new context, the newly created context would have to satisfy some nontrivial constraints and therefore could not be *freely* assigned.

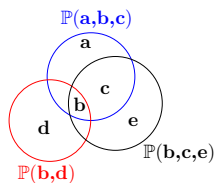


Figure 4: MCM for  $\mathbb{P}(\mathbf{a}, \mathbf{b}, \mathbf{c})$ ,  $\mathbb{P}(\mathbf{b}, \mathbf{d})$ , and  $\mathbb{P}(\mathbf{b}, \mathbf{c}, \mathbf{e})$ .

Let us discuss one final case to further clarify the process. Consider the multi-context model in Fig. 4. Could this model be constructed freely and gradually in a probabilistically consistent manner? The answer is positive. We first assign  $\mathbb{P}(\mathbf{a}, \mathbf{b}, \mathbf{c})$ , then we assign  $\mathbb{P}(\mathbf{b}, \mathbf{c}, \mathbf{e}) = \mathbb{P}(\mathbf{b}, \mathbf{c})\mathbb{P}(\mathbf{e}|\mathbf{b}, \mathbf{c})$  where  $\mathbb{P}(\mathbf{b}, \mathbf{c})$  is induced by our first assignment of  $\mathbb{P}(\mathbf{a}, \mathbf{b}, \mathbf{c})$ . Finally, we assign  $\mathbb{P}(\mathbf{b}, \mathbf{d}) = \mathbb{P}(\mathbf{b})\mathbb{P}(\mathbf{d}|\mathbf{b})$  where  $\mathbb{P}(\mathbf{b})$  is induced by our first assignment of  $\mathbb{P}(\mathbf{a}, \mathbf{b}, \mathbf{c})$ . A closer look reveals that this is not the only way we can gradually construct a contradiction-free model in this case: we could have performed the assignments in a different order<sup>5</sup>. Of course, the only thing which would have been different would be the induced probabilities. That is, if one does the assignment in the following

<sup>4</sup> $\mathbb{P}(\mathbf{y})$  is induced by the assignment of  $\mathbb{P}(\mathbf{x}, \mathbf{y})$ .

<sup>5</sup>Yet, this is not always the case: suppose there are four RVs in the domain, namely,  $\mathbf{a}$ ,  $\mathbf{b}$ ,  $\mathbf{c}$  and  $\mathbf{d}$  and we would like to assign  $\mathbb{P}(\mathbf{a}, \mathbf{b})$ ,  $\mathbb{P}(\mathbf{b}, \mathbf{c})$ , and  $\mathbb{P}(\mathbf{c}, \mathbf{d})$ . Performing the assignments in the order (1) –  $\mathbb{P}(\mathbf{a}, \mathbf{b})$ , (2) –  $\mathbb{P}(\mathbf{b}, \mathbf{c})$ , (3) –  $\mathbb{P}(\mathbf{c}, \mathbf{d})$  would not introduce any inconsistencies, in contrast to using the order (1) –  $\mathbb{P}(\mathbf{a}, \mathbf{b})$ , (2) –  $\mathbb{P}(\mathbf{c}, \mathbf{d})$ , (3) –  $\mathbb{P}(\mathbf{b}, \mathbf{c})$ .

order: (1)– $\mathbb{P}(\mathbf{b}, \mathbf{d})$ , (2)– $\mathbb{P}(\mathbf{a}, \mathbf{b}, \mathbf{c})$ , (3)– $\mathbb{P}(\mathbf{b}, \mathbf{c}, \mathbf{e})$  then the first assignment of  $\mathbb{P}(\mathbf{b}, \mathbf{d})$  will induce  $\mathbb{P}(\mathbf{b})$  for the second assignment of  $\mathbb{P}(\mathbf{a}, \mathbf{b}, \mathbf{c}) = \mathbb{P}(\mathbf{b})\mathbb{P}(\mathbf{a}, \mathbf{c}|\mathbf{b})$  and the second assignment will induce  $\mathbb{P}(\mathbf{b}, \mathbf{c})$  for the third assignment  $\mathbb{P}(\mathbf{b}, \mathbf{c}, \mathbf{e}) = \mathbb{P}(\mathbf{b}, \mathbf{c})\mathbb{P}(\mathbf{e}|\mathbf{b}, \mathbf{c})$ .

## 4 INFERENCE IN MCMS

In this section we consider *evidential* inference problems in multi-context settings. The objective is to evaluate (to the extent possible) a probability of the form  $\mathbb{P}(\mathbf{O} = O|\mathbf{E} = E)$ , called a *query*, where  $\mathbf{O}$  and  $\mathbf{E}$  are two mutually exclusive sets of RVs. The set  $\mathbf{E}$  is the set of evidence variables and  $\mathbf{O}$  is the set of RVs for which we are interested in knowing with what probability they take on the value  $O$ , upon the observation of  $\mathbf{E} = E$ . In multi-context settings, inference problems can be categorized into two broad classes:

- Intra-Contextual Inference Problems: For which the sets  $\mathbf{E}$  and  $\mathbf{O}$  both belong to the same context.
- Inter-Contextual Inference Problems: For which the sets  $\mathbf{E}$  and  $\mathbf{O}$  do not belong to a single context and, therefore, more than one context is involved in the inference problem.

In what follows, we will elaborate on these two cases.

### 4.1 INTRA-CONTEXTUAL INFERENCE PROBLEM

One advantage of MCMs is that, once an inference problem is found to be an intra-contextual inference problem, one can take advantage of the rich independence structure potentially governing the context to accomplish the task of inference in a computationally efficient way. For instance, if the probabilistic knowledge of a context is presented in a form of a BN, then one can benefit from a variety of exact or approximate methods already developed for BNs. For a comprehensive study of such methods the reader is referred to (Koller and Friedman 2009). Hence, it is of great interest to have contexts whose probabilistic knowledge can be represented in some form of a PGM with sufficiently rich independence structure for which inference problems can be solved in a computationally efficient way. For example, if the probabilistic knowledge of a context is to be modeled according to some BN, we would like that BN to be as sparsely connected as possible and enjoy low tree-width to ensure computational efficiency for the task of inference (Chandrasekaran, Srebro, and Harsha 2008).

### 4.2 INTER-CONTEXTUAL INFERENCE PROBLEM: INFERENCE GRAMMAR

In this section, we turn our attention to the task of inter-contextual inference. The RVs involved in the query for

the inter-contextual inference problem do not belong to a single context. For this reason, the answer to the query is inevitably in the form of an interval indicating a lower and upper bound for the query. Since  $\mathbb{P}(E|O) + \mathbb{P}(\bar{E}|O) = 1$  we have  $\mathbb{P}(E|O)_\uparrow = 1 - \mathbb{P}(\bar{E}|O)_\downarrow$ . Therefore, we can focus our attention on the minimization problem (i.e., identifying a lower bound to the probability of interest) realizing that any maximization problem (i.e., identifying an upper bound to the probability of interest) could be cast as a minimization problem and vice versa.

First, we are going to consider some simple queries which are posed to some example MCMs. These MCMs are depicted in Fig. 5(a-c). The goal here is to develop some insight as to which variables are indeed relevant and which are deemed irrelevant for a given query and the corresponding MCM.

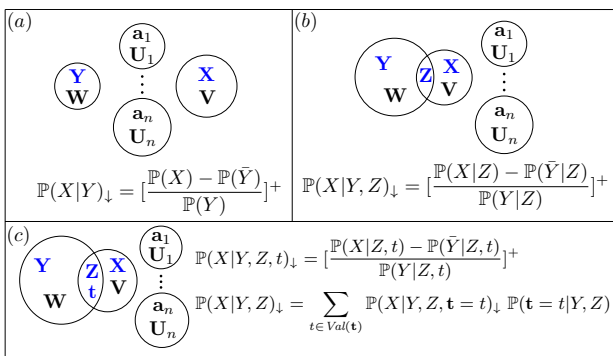


Figure 5: Sample inference rules given for some inter-contextual inference problems. The RVs involved in the query are shown in blue.

We begin by considering a simple case: the disjoint MCM shown in Fig. 5(a). The rule to evaluate  $\mathbb{P}(X|Y)_\downarrow$  is also given in Fig. 5(a). Interestingly enough, the expression only requires the intra-contextual quantities  $\mathbb{P}(X)$  and  $\mathbb{P}(Y)$  and it does not depend on any other RV present in the domain. In other words, as far as  $\mathbb{P}(X|Y)_\downarrow$  is concerned, the MCM shown in Fig. 5(a) is equivalent to a much simpler MCM: the one corresponding to having only two disjoint contexts described by  $\mathbb{P}(\mathbf{X})$  and  $\mathbb{P}(\mathbf{Y})$ . Next, we take the MCM given in Fig. 5(b) where there is an overlap between the context containing  $\mathbf{X}$  and the one containing  $\mathbf{Y}$ . The overlapping part consists of the random vector  $\mathbf{Z}$ . The rule to evaluate  $\mathbb{P}(X|Y, Z)_\downarrow$  is given in Fig. 5(b). Now, consider the MCM shown in Fig. 5(c) where we have the same setting we had in previous case but a new random variable  $\mathbf{t}$  is added in the overlapping region. Notice that the expression for  $\mathbb{P}(X|Y, Z, t)_\downarrow$  given in Fig. 5(c) is the same expression given for  $\mathbb{P}(X|Y, Z)_\downarrow$  in Fig. 5(b) with the substitution of  $Z, t$  instead of  $Z$ . That is,  $Z$  in Fig. 5(b) and  $Z, t$  in Fig. 5(c) are representing the same thing, namely, “all the variables in the overlapping region”, and in that respect, they are ultimately the same. The rules are

very much like sentences in predicate logic for which variables merely serve as place-holders.

The derivation of the rules given in Fig. 5(a-c) is not presented here. However, using the proof presented in Sec. A-II of Appendix (to identify the relevant variables) and subsequently following the methodology outlined in Sec. A-III of Appendix (to visualize the partitions and reason out the extent they overlap) it should be straightforward to derive the presented rules.

The sample set of rules presented is by no means exhaustive, nonetheless, due to the idea of context transformation that will be discussed in Sec. 4.3, they can be applied to a wide range of interesting inter-contextual inference problems. We would like to clarify that our ultimate objective is *not* to compute and provide the complete set of rules that can answer all possible queries and for all possible MCMs, since simply, the set is infinite in size. What we need, therefore, is an algorithm, let us call it  $\mathcal{I}^*$ , that can provide the answer to the posed query being given an MCM as an input. The presented rules provide insights and hints to the nature of  $\mathcal{I}^*$  which needs to be devised to ideally handle *any* arbitrary query posed to *any*<sup>6</sup> MCM. In a sense, we can get a glimpse of the nature of  $\mathcal{I}^*$  through analyzing the presented rules. In other words, the derived rules serve as a lens through which one can study  $\mathcal{I}^*$ . In Sec. A-I of Appendix a simple version of  $\mathcal{I}^*$  that can handle arbitrary MCMs is outlined.

The motivation behind giving this sample set of rules can now be summarized in the following.

1. To shed light on the general nature of a rule (which reflects on the nature of  $\mathcal{I}^*$ ). More specifically, to illustrate that a rule enjoys two key properties, namely: (i) scale-invariance, (ii) resemblance to sentences in predicate logic, in that in both cases, variables are mere place-holders. For this resemblance we refer to  $\mathcal{I}^*$  as *inference grammar*.
2. To demonstrate that a rule is telling us which intra-contextual quantities are essential and which are irrelevant for a particular inter-contextual query.
3. To emphasize the key property that a rule derived under a specific MCM remains valid for and can be applied to infinitely many other MCMs all of which are linked through the notions of nestedness and transformation; hence generalization is achieved.
4. To lay down the foundation of *transformation* and *nestedness* which both play crucial roles in understanding the underlying machinery behind  $\mathcal{I}^*$ .

<sup>6</sup>Although we believe that the MCMs generated through the generative process outlined in Sec. 3.2 are more cognitively plausible, nonetheless, from a pure mathematical point of view, it would be of interest to find an algorithm which could handle *any* MCM.



Next, we discuss another key property of the inference rules, namely, that of *scale-invariance*. Consider once again the case in Fig. 2. Now let us derive  $\mathbb{P}(x_i|y)_\downarrow$ , and  $\mathbb{P}(X|y)_\downarrow$  where  $\mathbf{X} \triangleq \mathbf{x}_{1:n}$ . Using the rule given in Fig. 5(a), one arrives at the following results:  $\mathbb{P}(x_i|y)_\downarrow = \left[\frac{\mathbb{P}(x_i) - \mathbb{P}(\bar{y})}{\mathbb{P}(y)}\right]^+$ , and  $\mathbb{P}(X|y)_\downarrow = \left[\frac{\mathbb{P}(X) - \mathbb{P}(\bar{y})}{\mathbb{P}(y)}\right]^+$ . In other words, the expressions remain the same, regardless of the dimension of the quantity of interest, i.e., be it a single RV or be it a random vector comprised of many RVs. In this respect, once again, the inference rules resemble expressions in predicate logic. The intuition on the scale invariance is provided in Sec. A-III of Appendix.

It is worth noting that  $\mathcal{I}^*$  formulates the inter-contextual inference problem as a Linear Programming (LP) optimization (cf. Sec. A-I of Appendix). The key issues to consider are: (i) what RVs have to be included in the LP, and (ii) the abstraction level  $\mathcal{I}^*$  should choose to encode the RVs identified in step (i) for the LP, i.e., the parametrization of RVs identified in step (i) for the LP. In what follows, the concepts of nestedness and transformation are put forth. Once the two are introduced, one could apply a single rule (e.g., one in Fig. 5(a)) to a much larger number of MCMs; in fact to infinitely many MCMs.

### 4.3 INTER-CONTEXTUAL INFERENCE PROBLEM: NESTEDNESS AND TRANSFORMATION

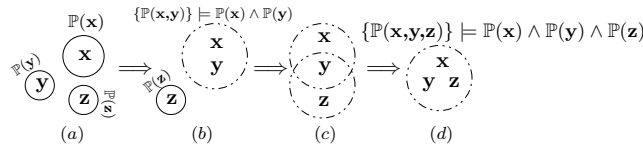


Figure 6: Inter-Contextual Inference Problem: Transformation and hierarchical construct. As one proceeds from the left to the right, a more comprehensive knowledge of domain is assumed to be available, of course hypothetically.

The nested property, or *nestedness*, refers to the fact that every MCM can be considered as an element of a family of MCMs. That family contains all MCMs which through marginalization can produce the original MCM. In such a case we simply say that the nested property holds between the original MCM and the family. The process of going from the original MCM to one of the members of the family is referred to as *transformation*. For example, the MCM containing three contexts  $\{\mathbf{x}\}$ ,  $\{\mathbf{y}\}$ , and  $\{\mathbf{z}\}$  shown in Fig. 6(a) is a member of a family of MCMs containing two contexts  $\{\mathbf{x}, \mathbf{y}\}$  and  $\{\mathbf{z}\}$ , shown in Fig. 6(b), one of which is associated to a *family* of JPDs over  $\mathbf{x}$  and  $\mathbf{y}$  (the dash-dotted circle in Fig. 6(b)) which, if marginalized, produces the same  $\mathbb{P}(\mathbf{x})$  and  $\mathbb{P}(\mathbf{y})$  in the original MCM (left-most MCM). Mathematically, the set of all

JPDs over RVs  $\mathbf{x}$  and  $\mathbf{y}$  which, if marginalized, produce specific marginal probability distributions  $\mathbb{P}(\mathbf{x})$  and  $\mathbb{P}(\mathbf{y})$  is denoted by  $\{\mathbb{P}(\mathbf{x}, \mathbf{y})\} \models \mathbb{P}(\mathbf{x}) \wedge \mathbb{P}(\mathbf{y})$ . The notion of the nested property enables us to look at one MCM as a subset of another larger MCM. The nested property, furthermore, enables one to sort MCMs in a hierarchical construct as illustrated in Fig. 6 where moving from the left to the right corresponds to moving from lower levels of hierarchy to higher levels.

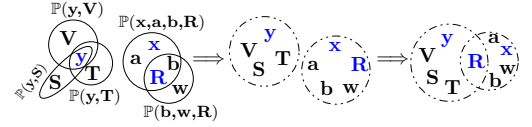


Figure 7: Transformation: Sample case.

To convey the idea, consider the case illustrated in Fig. 7. Suppose the query of interest is  $\mathbb{P}(x|y, R)_\downarrow$ . Then, one can first transform the original (left-most) MCM into the MCM shown in the middle, and subsequently into the right-most MCM. Hence, using the right-most MCM and the rule given in Fig. 5(b), one can write  $\mathbb{P}(x|y, R)_\downarrow = \left[\frac{\mathbb{P}(x|R) - \mathbb{P}(\bar{y}|R)}{\mathbb{P}(y|R)}\right]^+ = \left[\frac{\mathbb{P}(x|R) - 1 + \mathbb{P}(y|R)}{\mathbb{P}(y|R)}\right]^+$ . If we had the knowledge of  $\mathbb{P}(y|R)$  then the expression given above would have been sufficient to derive  $\mathbb{P}(x|y, R)_\downarrow$ . However, since  $\mathbb{P}(y|R)$  is *not* known, we need to go through one more step. This is precisely due to, and emphasizes, the fact that by working on the right-most MCM we implicitly presumed that we were equipped with more knowledge than we really had. Using the middle MCM and the rule given in Fig. 5(a), one can conclude  $\mathbb{P}(y|R)_\downarrow = \left[\frac{\mathbb{P}(y) - \mathbb{P}(R)}{\mathbb{P}(R)}\right]^+$ . Altogether<sup>7</sup>,  $\mathbb{P}(x|y, R)_\downarrow = \left(\left[\frac{\mathbb{P}(x|R) - 1 + \mathbb{P}(y|R)}{\mathbb{P}(y|R)}\right]^+\right)_\downarrow = \left[\frac{\mathbb{P}(x|R) - 1 + \mathbb{P}(y|R)_\downarrow}{\mathbb{P}(y|R)_\downarrow}\right]^+$ . It is worth noting that the same rule would apply if instead of the random vector  $\mathbf{R}$  we were dealing with the random variable  $\mathbf{a}$ , i.e., to find  $\mathbb{P}(x|y, a)_\downarrow$  one could use the same expression given for  $\mathbb{P}(x|y, R)_\downarrow$  by substituting  $a$  in place of  $R$  in all the expressions. Arguments of this kind are made possible due to the idea of transformation which enables us to analyze the transformed MCM (e.g., the middle one in Fig. 7) rather than the original MCM (the left-most one in Fig. 7). Furthermore, the concept of transformation highlights a key idea: if a piece of information (i.e., an intra-contextual quantity) is irrelevant in the transformed MCM for the posed query, it must have been irrelevant in the original MCM in the first place. This statement, once again, sheds light on what intra-contextual quantities are relevant or irrelevant to derive a posed inter-contextual query on a given MCM.

<sup>7</sup>This is due to the observation that for function  $f(y) = \left(\frac{k+y}{y}\right)$  when  $k < 0$ ,  $\min_{1 \geq y \geq t > 0} f(y) = \left(\frac{k+t}{t}\right)$ .

## 5 DISCUSSION

We will now discuss related work so as to build a connection between ours and previous attempts to incorporate partial probabilistic knowledge of a domain in the task of inference.

Attempting to combine Probabilistic Logic and BNs, the authors in (Andersen and Hooker 1990; 1994) formulate the inference problem as an optimization problem subject to non-linear constraints so as to incorporate the conditional independence relations embedded in the BN. However, in our proposed framework, the issue of dealing with conditional independence relations does not arise at all, because these relations are dealt with during the derivation process of intra-contextual probabilities.

The authors of (Hansen et al. 1995) point out that one could avoid non-linear optimization when the value for a conditional probability is at least imprecisely known. For example, the constraint  $\mathbb{P}(a|b) = \mathbb{P}(a)$ , if the value for  $\mathbb{P}(a)$  is known either precisely or imprecisely within some interval  $[\alpha, \beta]$ , can be written as

$$\frac{\mathbb{P}(a, b)}{\mathbb{P}(b)} = \mathbb{P}(a) \in [\alpha, \beta] \Leftrightarrow \begin{cases} \mathbb{P}(a, b) - \alpha\mathbb{P}(b) > 0, \\ \mathbb{P}(a, b) - \beta\mathbb{P}(b) < 0. \end{cases}$$

Hence, the independence  $\mathbb{P}(a|b) = \mathbb{P}(a)$  can be formulated as a number of linear constraints. However, the main drawback of this approach is that encoding a conditional independence relation such as  $\mathbb{P}(\mathbf{x}|\mathbf{y}, \mathbf{a}_1, \dots, \mathbf{a}_n) = \mathbb{P}(\mathbf{x}|\mathbf{y})$  requires a number of linear equations that is exponential in  $n$  to be introduced into the optimization problem (Andersen and Hooker 1994).

Drawing on the idea of Context-Specific Independence (CSI) (Boutilier et al. 1996), the authors of (Geiger and Heckerman 1991) propose the Bayesian Multinet model which aims at taking advantage of the existing CSIs to perform inference, by modeling a single BN as multiple context-specific BNs. Translated into our multi-context setting, the Bayesian Multinet model corresponds to the case where the whole domain is modeled as a single BN, i.e., a single-context MCM, that can be decomposed into multiple BNs each being valid for a specific instantiation of some RVs in the domain.

The authors of (Thone, Guntzer, and Kiebling 1992) point out the same concerns which led us to propose MCM, namely: (i) If unverified (in)dependencies are imposed between the variables in the domain then implausible results may arise; (ii) PGMs require one to have complete probabilistic knowledge of a domain which may not be available. Motivated by these, (Thone, Guntzer, and Kiebling 1992) gives a collection of rules to carry out inference in a domain. Broadly speaking, this work is similar to ours in spirit with the main distinction being the level of abstraction chosen to perform inference. In (Thone, Guntzer, and

Kiebling 1992) inference is performed in a very local and rule-based fashion and conditional independence relations are dealt with directly which complicates the task at hand; a task which is futile when it comes to dealing with domains of many variables. In our case, by introducing the notion of context and encoding conditional independence relations within contexts we avoid having to contemplate the intra-contextual inference problem and leave this task for the corresponding context. This way, we can take advantage of the possibly rich independence structure governing the context and carry out the intra-contextual inference problem in a computationally efficient manner.

Finally, let us discuss some interesting aspects of the proposed model.

The degree of belief is encoded mathematically in the form of a probability distribution over the variables contained within the context. Furthermore, in the process of partial belief formation (which leads to the formation of contexts) the reasoner is ignorant as to how various contexts probabilistically interact (are related), except that, some contexts may in fact share a number of variables in between and hence overlap. Later on, in the process of the derivation of the query posed to the reasoner, this ignorance manifests in the uncertainty region represented by the min/max values for the inter-contextual query of interest. In other words, if the reasoner incurs ignorance as to the (in)dependency structure governing the variables present in the domain, then later on, in the process of derivation of the posed query, the reasoner has to pay the price by merely arriving at a *probability interval* rather than a point probability as an answer to the query of interest. Yet, the knowledge of the underlying dependency structure is a fundamental knowledge whose availability to the reasoner should *not* be postulated as an inevitability but as an advantaged position.

The evolutionary process of MCM does not enforce a specific gradual expansion path, for the claim of MCM is merely that *any partial belief formation as to the domain can be modeled in the framework depicted by MCM*. That is, the reasoner may arrive at different MCMs, depending on the order in which the reasoner encounters different concepts and also depending on her background knowledge as to the nature of the potential connections between a collection of variables. Simply put, the order according to which the reasoner comes about knowing the concepts or propositions of the domain does matter (cf. the discussion on the order of belief formation in Sec. 3.2).

MCM enables one to carry out inference without having to commit to any unjustified or uncertain independence assumptions. In light of this, contexts symbolize the regions of the domain over which an (in)dependence structure is presumed and hence, the growth and merging of contexts indicates the formation of new (in)dependence structures over some parts of the domain which previously were un-

structured. In short, MCM is meant to be invoked in circumstances where the observations and the a priori knowledge combined are not sufficient for the reasoner to form the full JPD over all of the domain variables and yet, quite crucially, the reasoner is reluctant to submit to any unjustified assumptions to compensate for such inadequacy of knowledge.

## 6 CONCLUSION

In an attempt to establish a middle ground between Bayesian Logic and Probabilistic Logic (Andersen and Hooker 1990; 1994), on one side, and PGMs<sup>8</sup> on the other, we proposed the Multi-Context Model to represent the state of partial knowledge regarding a domain. The generative process for the gradual construction of contradiction-free MCMs was discussed. The task of Inference for MCM was studied and, along the path, the notions of inference grammar, nestedness, and transformation were introduced. A short version of  $\mathcal{I}^*$  without the scale-invariance property was provided in Appendix. It is worth noting that scale-invariance property can be achieved with a minor change to the last step of the proposed algorithm.

## APPENDIX

### A-I $\mathcal{I}_{non-scale}^*$ : A short version of $\mathcal{I}^*$ without scale-invariance property

$\mathcal{I}^*$  aims at minimally parameterizing the information contained in an MCM so that the posed inter-contextual query can be stated as an LP with the fewest number of parameters. As pointed out earlier in Sec. 4.2,  $\mathcal{I}^*$  has to decide on the following: (i) what RVs have to be included in the LP, and (ii) the abstraction level required to minimally encode the information on the RVs identified in step (i) for the LP, in our case, the parametrization of the identified RVs.

In what follows, a simple algorithm,  $\mathcal{I}_{non-scale}^*$ , is sketched which only performs (i) and ignores (ii). In other words,  $\mathcal{I}_{non-scale}^*$  identifies the relevant RVs needed to derive the *exact* lower/upper bound for the inter-contextual query, however, it does not aim at minimally encoding them into the LP<sup>9</sup>.  $\mathcal{I}_{non-scale}^*$  consists of three steps:

(1) Identify all the RVs involved in the posed query (e.g., in  $P(X|Y, z)$  these are the random vector  $X$ , random vector  $Y$  and RV  $z$ ).

(2a) If any two of the already identified RVs belong to two

<sup>8</sup>For instance, Bayesian Networks (Pearl 1986), Markov Networks (Koller and Friedman 2009), and Chain Graphs (Buntine 1995).

<sup>9</sup>To read more on this, the reader is referred to the discussion on scale-invariance property in Sec. 4.2 and Sec. A-III of Appendix.

overlapping contexts, identify all the *overlapping* RVs between these two contexts (e.g., in Fig. 5(b) and for the query  $P(X|Y)$  for which step (1) would identify  $X$  and  $Y$ , random vector  $Z$  in the overlapping region must be identified as well).

(2b) If any two of the already identified RVs belong to two contexts connected through a chain of overlapping contexts: identify all the RVs contained in all the *overlapping* regions of the chain of contexts.

(3) Parameterize only the identified RVs in steps (1), (2a), and (2b) (remove all the other RVs from the MCM—there is no need to encode the information on any other RVs not identified in steps (1), (2a), and (2b)).

■

It should be noted that whether the posed query involves minimization or maximization does not affect which RVs need to be identified by  $\mathcal{I}_{non-scale}^*$ . Finally, It is worth noting that with a minor modification to step (3) of  $\mathcal{I}_{non-scale}^*$ , the scale-invariance property could be achieved. The modification has to do with the question of how to *minimally* encode the information on each RV identified in steps (1), (2a), and (2b) of  $\mathcal{I}_{non-scale}^*$ .

To demonstrate the operation of  $\mathcal{I}_{non-scale}^*$  on a more complicated MCM that involves loops, consider the following example sketched in Fig. 8(a). The query of interest is  $\mathbb{P}(X|Y)_{\downarrow}$ .

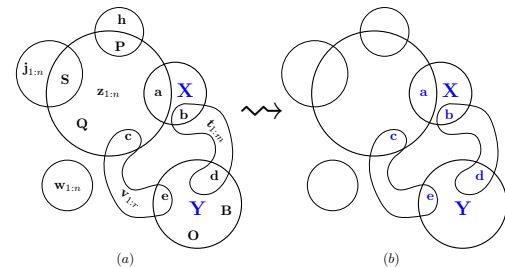


Figure 8: (a) Sample MCM. The RVs involved in the posed query are depicted in blue. (b) In Step (1)  $X$  and  $Y$  are identified; in step (2b) the RVs  $b$ ,  $d$  as well as  $a$ ,  $c$ , and  $e$  are identified. According to step (3) of  $\mathcal{I}_{non-scale}^*$  all of the information as to the RVs  $X$ ,  $Y$ ,  $b$ ,  $d$ ,  $a$ ,  $c$ , and  $e$  has to be stated as an LP to derive the query.

Next, we are going to sketch the proof for  $\mathcal{I}_{non-scale}^*$ . Let us first state the claim formally and then provide the proof.

### A-II Proof for $\mathcal{I}_{non-scale}^*$ :

**Lemma:** *Given a posed query and an MCM, if all the information on the RVs identified in steps (1) to (2b) of  $\mathcal{I}_{non-scale}^*$  is stated and then solved as an LP, the exact solution (i.e., a min or max) can be derived for the posed*

query; all the remaining information available in the MCM is deemed irrelevant to the derivation of the query, hence the sufficiency.

**Proof:** Our proof is constructive. In the proof we entertain two ideas, namely (i) the idea of generative process and, particularly, that of *conditioning* also used in Sec. 3.2, and (ii) the notion we refer to as the *locality of information*. Suppose that all the RVs discussed in steps (1) to (2b) of  $\mathcal{I}_{non-scale}^*$  are identified. The key insight is that the information on how the remaining RVs probabilistically interact with each other is completely local in nature and, therefore, irrelevant to the derivation of the posed query. To see this, one can start off with the identified RVs and then in a gradual fashion add on<sup>10</sup> the rest of the RVs (through the idea of conditioning discussed in Sec. 3.2). Quite crucially, this very process of adding the non-identified RVs to the model can be done completely in a local fashion, i.e., without imposing any constraints on how the identified RVs probabilistically interact. The mere fact that those RVs can be added into the model: (i) subsequent to the identified ones, and (ii) without inducing any sort of constraints on the identified ones, deems them irrelevant to the derivation of the query. ■

### A-III Scale-Invariance Property: Intuition

Here, we will provide a proof for the example on scale-invariance property given in Sec. 4.2. Although the proof is provided for a special query, the methodology used in the proof provides an insightful way of *visualizing* an inference problem. The idea behind the proof is very simple and related to visualizing the connection of a RV to the underlying sample space using Venn diagrams. Without loss of generality, we assume that all the RVs present in the domain are binary<sup>11</sup>. Random vector  $\mathbf{X} = \mathbf{x}_{1:n}$  partitions the sample space  $\Omega$  into  $2^n$  disjoint regions each of which corresponds to a realization of  $\mathbf{X}$ . If each realization of the random vector  $\mathbf{x}_{1:n}$  corresponds to a binary number (i.e., binary-coding the realizations), then one can conclude  $Val(\mathbf{X}) = \{0, 1, \dots, 2^n - 1\}$ . Let us index the partitions by their corresponding realization of  $\mathbf{X}$ . An illustrative example of an induced partitioning of the sample space  $\Omega$  due to random vector  $\mathbf{X} = \mathbf{x}_{1:n}$  is depicted in Fig. 9(a), and a partitioning induced by RVs  $\mathbf{y}$  and  $\mathbf{z}$  is sketched in Fig. 9(b). We note that the mere knowledge of the distribution function of a random quantity does not provide one with the knowledge of the underlying partitions. For this particular example, since the JPD over  $\mathbf{X}, \mathbf{y}, \mathbf{z}$  is not available, the knowledge of how the partitions induced by  $\mathbf{y}, \mathbf{z}$  (Fig. 9(b)) and the ones induced by  $\mathbf{X}$  (Fig. 9(a)) interact, i.e., to what extent they overlap,

remains unspecified. Therefore, since  $\mathbb{P}(X|y) = \frac{\mathbb{P}(X,y)}{\mathbb{P}(y)}$ , to minimize (maximize)  $\mathbb{P}(X|y)$ , the quantity  $\mathbb{P}(X, y)$  has to be minimized (maximized). Pictorially, the minimization (maximization) of  $\mathbb{P}(X, y)$  corresponds to the minimization (maximization) of the overlap between the partitions corresponding to the events  $\{\mathbf{X} = X\}$  and  $\{\mathbf{y} = y\}$ ; hence, very simply,  $\mathbb{P}(X, y)_\downarrow = [\mathbb{P}(X) + \mathbb{P}(y) - 1]^+$  and  $\mathbb{P}(X, y)_\uparrow = \min\{\mathbb{P}(X), \mathbb{P}(y)\}$ . The key point, which yields the scale-invariance property, is that to derive the minimum (maximum) overlap between the partitions corresponding to the events  $\{\mathbf{X} = X\}$  and  $\{\mathbf{y} = y\}$  the information as to how the other partitions—corresponding to the other realizations of the present RVs in the model—interact with one another neither needs to be known nor to be encoded into the LP; a fact which results in not requiring to encode the information as to the other realizations. Hence the only pieces of information that are required to be encoded and then solved as an LP are  $\mathbb{P}(X)$  and  $\mathbb{P}(y)$ . The same line of reasoning could be adopted for  $\mathbb{P}(x_i|y)$ . The idea of scale-invariance, therefore, aims to avoid the encoding of the information as to the partitions induced on  $\Omega$  which are yet deemed to be irrelevant to the derivation of the posed query; hence one needs to encode solely the relevant ones into the LP.

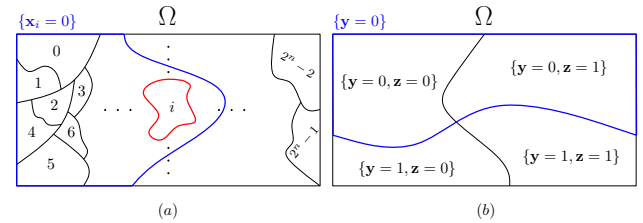


Figure 9: Sample Space: (a) Partitioning induced on  $\Omega$  due to  $\mathbf{X} = \mathbf{x}_{1:n}$ . The blue region corresponds to the partition associated to the event  $\{\mathbf{x}_i = 0\}$  and the red one to that of  $\{\mathbf{X} = i\}$  where  $i \in Val(\mathbf{X})$ . (b) Partitioning induced on  $\Omega$  due to RVs  $\mathbf{y}$  and  $\mathbf{z}$ . The blue region corresponds to the partition associated to the event  $\{\mathbf{y} = 0\}$ .

### Acknowledgement

The authors would like to thank the anonymous reviewers for their valuable comments.

This work was supported in part by the Natural Sciences and Engineering Research Council (NSERC) under grant RGPIN 262017 and by the Fonds Quebecois de la Recherche sur la Nature et les Technologies (FQRNT).

<sup>10</sup>This is based on the fundamental property that a JPD can be expanded using the chain rule of probability in an arbitrary order.

<sup>11</sup>The generalization of the argument to non-binary RVs is straightforward.

## References

- Andersen, K. A., and Hooker, J. N. 1990. Probabilistic logic for belief nets. In *International Congress of Cybernetics and Systems*. New York City.
- Andersen, K. A., and Hooker, J. N. 1994. Bayesian logic. *Decision Support Systems* 11(2):191–210.
- Boutilier, C.; Friedman, N.; Goldszmidt, M.; and Koller, D. 1996. Context-specific independence in bayesian networks. 115–123. Morgan Kaufmann Publishers Inc.
- Buntine, W. L. 1995. Chain graphs for learning. In *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, 46–54. Morgan Kaufmann Publishers Inc.
- Chalmers, A. F. 1976. *What is this thing called science?* Hackett Publishing.
- Chandrasekaran, V.; Srebro, N.; and Harsha, P. 2008. Complexity of inference in graphical models. *Proceedings of the 24th Conference in Uncertainty in Artificial Intelligence* 70–78.
- Geiger, D., and Heckerman, D. 1991. Advances in probabilistic reasoning. In *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*, 118–126. Morgan Kaufmann Publishers Inc.
- Hansen, P.; Jaumard, B.; Nguetse, G. D.; and Aragao, M. P. D. 1995. *Models and algorithms for probabilistic and Bayesian logic*. Citeseer.
- Koller, D., and Friedman, N. 2009. *Probabilistic graphical models: principles and techniques*. MIT press.
- Pearl, J. 1985. Bayesian networks: a model of self-activated memory for evidential reasoning. in *Proceedings of the Cognitive Science Society* 329–334.
- Pearl, J. 1986. Fusion, propagation, and structuring in belief networks. *Artificial intelligence* 29(3):241–288.
- Pearl, J. 1990. Reasoning with belief functions: An analysis of compatibility. *International Journal of Approximate Reasoning* 4(5):363–389.
- Thone, H.; Guntzer, U.; and Kiebling, W. 1992. Towards precision of probabilistic bounds propagation. In *Uncertainty in Artificial Intelligence*, 315–322.

---

# Annealed Gradient Descent for Deep Learning

---

Hengyue Pan Hui Jiang

Department of Electrical Engineering and Computer Science  
York University, 4700 Keele Street, Toronto, Ontario, Canada  
Emails: panhy@cse.yorku.ca hj@cse.yorku.ca

## Abstract

Stochastic gradient descent (SGD) has been regarded as a successful optimization algorithm in machine learning. In this paper, we propose a novel annealed gradient descent (AGD) method for non-convex optimization in deep learning. AGD optimizes a sequence of gradually improved smoother mosaic functions that approximate the original non-convex objective function according to an annealing schedule during the optimization process. We present a theoretical analysis on its convergence properties and learning speed. The proposed AGD algorithm is applied to learning deep neural networks (DNNs) for image recognition on MNIST and speech recognition on Switchboard. Experimental results have shown that AGD can yield comparable performance as SGD but it can significantly expedite training of DNNs in big data sets (by about 40% faster).

## 1 INTRODUCTION

The past few decades have witnessed the success of gradient descent algorithms in machine learning. By only calculating the local gradient information of the loss function, gradient descent (GD) algorithms may provide reasonably good optimization results for different types of problems. Among many, *stochastic gradient descent (SGD)* is a very popular method for modern learning systems, which only use one or a few randomly-selected training samples to update the model parameters in each iteration (Bottou, 1998). In comparison to the batch GD algorithms, SGD requires far less computing resources especially when it deals with a huge task involving big data. In (LeCun and Bottou, 2004), it has been proved that SGD can process asymptotically more training samples than batch GD algorithms given the same amount of computing resources. (Roux et al., 2012) proposed a new SGD framework that can achieve

a linear convergence rate for strongly-convex optimization problems. In (Shamir and Zhang, 2013), the performance of SGD was analyzed on a number of non-smooth convex objective functions and a bound on the expected optimization errors was provided. On the other hand, SGD also has its own drawbacks. The random noise introduced by data sampling leads to noisy gradient estimates, which may slow down the convergence speed and degrade the performance (Murata and Amari, 1999). Moreover, because of its sequential nature, SGD is hard to parallelize. More recently, several different methods have been proposed to parallelize SGD to accelerate its training speed for big-data applications. Initially, some researchers have proposed to implement the SGD method across multiple computing nodes that are synchronized for updating model parameters. Unfortunately, it has been found that the delay by the required server synchronization is always much longer than the time needed to calculate the gradient. Therefore, several other methods have been proposed to parallelize SGD without frequent synchronization among computing nodes. For example, Zinkevich et al. (2009) has presented a parallelized SGD algorithm, which dispatches all training samples into several computing nodes to update the parameters independently, and the final models will be combined by averaging all separate models at the end of each training epoch. Moreover, Bockermann and Lee (2012) have proved that the performance of this parallelized SGD algorithm depends on the number of SGD runs, and they have successfully used it to train large-scale support vector machines. However, the convergence of this simple parallelized SGD method requires that the learning process is convex. Moreover, Agarwal and Duchi (2012) have shown that the delays introduced by asynchronous model updates can be asymptotically neglected when we optimize a smooth and convex problem. Similarly, Feng et al. (2011) has proposed a parallelized SGD framework called *HOGWILD!*, which has mostly removed the memory locking and synchronization. However, it has found that this method works well only for sparse models. In summary, these parallelized SGD methods heavily rely on the assumptions that the learning problems are convex and/or sparse. These methods may suf-

for the performance degradation when dealing with more general non-convex optimization problems such as those in deep learning.

Recently, deep learning (Bengio, 2009) has achieved huge successes in many real-world applications, such as speech recognition and computer vision. It becomes a very interesting problem to learn large-scale deeply-structured neural networks, such as *deep neural networks (DNNs)*, from big data sets. We know that the training of DNNs is highly non-convex. Moreover, it is relatively expensive to compute the gradients of the objective function for DNNs since it needs to run the time-consuming back-propagation algorithm. To accelerate the large scale DNN training for big data, it has proposed a weight sharing method in (LeCun et al., 1989), which reduces the number of free parameters in the neural network and thus speeds up the training procedure. Even though today’s development of computing hardware makes it possible to train large DNNs directly, it is still very slow to train state-of-the-art DNNs for many real-world applications since the major training of DNNs still depends on the mini-batch SGD algorithm. Therefore, it is much needed in deep learning to develop new optimization methods that are faster to solve large-scale training problems in deep learning. One idea is ‘starting small’ (Elman, 1993), in which the network training will begin from simple training data with small working memory, and gradually increase the data complexity and network memory. This process simulates the learning procedure of human beings, especially in some complex domains like language. Krueger et al. have implemented the so-called ‘shaping’ learning in the neural network training (Krueger and Dayan, 2009). During the training, the task is split into several sub-components and a suitable training sequence is used to boost the training speed. In (Bengio et al., 2009), Bengio et al. have proposed a training strategy for deep learning called curriculum learning. The basic idea is to start the learning process from small tasks that are easy to solve, and gradually increase the complexity of the tasks in the later learning stage. Experimental results imply that when using a suitable curriculum, this training strategy may provide a similar performance as unsupervised pre-training and it helps the algorithm to find a better local minimum. The curriculum learning method can serve as an important basis for the work in this paper.

In this paper, we propose a new algorithm called *annealed gradient descent (AGD)*. Instead of directly optimizing the original non-convex objective function, the basic idea of AGD is to optimize a low resolution approximation function that may be smoother and easier to optimize. Furthermore, the approximation resolution is gradually improved according to an annealing schedule over the optimization course. In this work, we have proposed to approximate a non-convex objective function based on some pre-trained codebooks, where the approximation precision can be eas-

ily controlled by choosing different number of codewords in the codebook. In comparison with (Bengio et al., 2009), the main contribution of this paper is that AGD provides a suitable way for approximation (through pre-trained codebooks), and more importantly, we show a bound for the difference between the parameters derived by AGD and the regular GD algorithms. This new method has several advantages: Firstly, the low resolution approximation by codebooks lead to a much smoother risk function, which may result in finding a good local minimum more easily. Secondly, because the size of each codebook is much smaller than that of the training set, we can use a fast batch algorithm to learn the model at the beginning and this part can be easily parallelized. In this work, we have applied AGD to training DNNs for various tasks to verify its efficiency and effectiveness. Experiments have shown that the AGD algorithm yields about 40% speed-up in total training time of DNNs, and also leads to similar recognition performance as the regular mini-batch SGD.

The remainder of this paper is organized as follows. In section 2, we provide some background information about empirical risk function and two kinds of gradient descent algorithm. Section 3 shows the mosaic risk function and some related theoretical analysis. In section 4, we present the proposed AGD algorithm. Section 5 reports experiments on different tasks, and we conclude this paper in section 6.

## 2 PRELIMINARIES

In this section, we first review some preliminary definitions in machine learning, which serve as important notation bases for this work.

### 2.1 EMPIRICAL RISK FUNCTION

In machine learning, we normally use a *loss function*,  $Q(x, y, \theta)$ , to measure the ‘cost’ of a given event  $x$  ( $y$  is the corresponding label of  $x$ ) and the underlying model parameters are denoted as  $\theta$ , and the expected value of the loss function is the so-called *expected risk function*,  $R(\theta)$ :

$$R(\theta) = \mathbf{E}[Q(x, y, \theta)] \triangleq \int Q(x, y, \theta) dP(x, y) \quad (1)$$

where  $P(x, y)$  denotes the ground truth distribution over all possible events. The fundamental goal of many machine learning problems is to minimize the above expected risk function. In practice, however, it is extremely hard to do so because  $P(x, y)$  is always unknown. Therefore, In practice, we normally use a finite training set that includes  $N$  independent pairs of sample  $O_N = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$ , which are presumably randomly sampled from the above unknown distribution. Based on the training set, we may derive the so-called *empirical risk function*,  $R_N(\theta)$ , to approximate the expected

risk function in eq.(1):

$$R(\theta) \approx R_N(\theta) = \frac{1}{N} \sum_{n=1}^N Q(x_n, y_n, \theta) \quad (2)$$

If the training set is sufficiently large, under some minor conditions, minimizing the empirical risk function in eq.(2) may also minimize the expected risk function in eq.(1) (Vapnik, 1998). For notational clarity, without confusion, we drop label  $y_n$  from the loss function for the rest of this paper.

## 2.2 GRADIENT DESCENT ALGORITHM

To minimize the empirical risk function, we can use gradient descent algorithms, which update  $\theta$  along the direction of the negative gradient based on a pre-defined learning rate  $\lambda$ . Generally speaking, there are two different types of gradient descent algorithms: batch gradient descent (batch GD) and stochastic gradient descent (SGD).

In each iteration, the batch GD considers all of the training samples to calculate the average gradient and then update the parameters accordingly:

$$\begin{aligned} \hat{\theta}_{t+1} &= \hat{\theta}_t - \lambda_t \cdot \nabla_{\theta} R_N(\hat{\theta}_t) \\ &= \hat{\theta}_t - \lambda_t \cdot \frac{1}{N} \sum_{n=1}^N \frac{\partial Q(x_n, \hat{\theta}_t)}{\partial \theta} \end{aligned} \quad (3)$$

where  $\lambda_t$  is the learning rate at iteration  $t$ . In contrast, SGD only takes one training sample (which is randomly sampled from the training set) into account in each iteration:

$$\bar{\theta}_{t+1} = \bar{\theta}_t - \lambda_t \cdot \frac{\partial Q(x_n, \bar{\theta}_t)}{\partial \theta}. \quad (4)$$

If we set a suitable learning rate, under some conditions, both batch GD and SGD can finally converge to a local minimum  $\theta^*$  of the empirical risk function (Bottou, 2004). In practice, to reduce variance of the estimated gradients in SGD, a variant SGD, called mini-batch SGD, is normally used, where a small set (called mini-batch) of randomly selected data samples are used to estimate the gradient for each model update, as opposed to only one sample in SGD.

As we know, the batch GD works well for convex optimization while SGD may be used to solve non-convex optimization problems due to the random noises in its gradient estimation. Meanwhile, SGD requires far less computing resources in comparison to batch algorithms, but on the downside, its convergence speed is very slow due to sampling noise, and it is very hard to parallelize SGD. Therefore, when dealing with some large scale tasks, SGD may run very slowly. In this paper, we propose a new optimization method to solve some large-scale non-convex optimization problems in deep learning. The new method will be compared with SGD in terms of convergence speed and learning performance.

## 3 THE MOSAIC RISK FUNCTION

Some previous work has considered the problem of critical points (including local optima and saddle points) in non-convex optimizations. According to (Choromanska et al., 2014), some poor local minima may hinder the optimization process especially in small-scale neural networks. (Dauphin et al., 2014) argued that for the practical high dimensional problems, the saddle points may become the most difficult problem to deal with, rather than local optima. In practice, any local search algorithms may be easily trapped into a nearby shallow local optimum point or saddle point, which makes it hard for optimization to proceed further. SGD relies on the sampling noise to alleviate this problem. Another way to tackle this problem is to optimize a smoother approximation of the original rugged non-convex function. In this work, we propose to approximate the original objective function based on a relatively small codebook, which is generated by clustering the whole training set. In this way, we may provide a low resolution approximation of the objective function, which is much smoother and easier to optimize with simple local search algorithms.

Assume we use a discrete codebook, denoted as  $C = \{c_1, c_2, \dots, c_M\}$ , where  $M \ll N$ , to approximate the original training set. For a training sample  $x_n$ , we select its nearest codeword in  $C$  as its approximation:

$$c^{(n)} = \arg \min_{c_m \in C} \|x_n - c_m\| \quad (5)$$

and the quantization error  $\epsilon_n$  is  $\|x_n - c^{(n)}\|$ .

Next, we may derive a low resolution risk function  $\tilde{R}_{\epsilon}$ , called *mosaic risk function*, to approximate the empirical risk function  $R_N(\theta)$  (where  $\epsilon \equiv \max_n \epsilon_n$ ):

$$\tilde{R}_{\epsilon}(\theta) = \frac{1}{N} \sum_{n=1}^N Q(c^{(n)}, \theta) = \sum_{m=1}^M \frac{\omega_m}{N} \cdot Q(c_m, \theta) \quad (6)$$

where  $\omega_m$  is the number of training samples in the whole training set that are approximated by the codeword  $c_m$ , i.e.,  $\omega_m = \sum_{n=1}^N \delta(c^{(n)} - c_m)$ , where  $\delta(\cdot)$  denotes the Kronecker delta function.

Assume that the loss function  $Q(x, \theta)$  is twice Lipschitz-continuous with respect to the input sample  $x$  and the model parameter  $\theta$ , that is,

$$\|Q(x_i, \theta) - Q(x_j, \theta)\| < L_0 \|x_i - x_j\| \quad (7)$$

$$\|Q'(x_i, \theta) - Q'(x_j, \theta)\| < L_1 \|x_i - x_j\| \quad (8)$$

$$\|Q(x, \theta_i) - Q(x, \theta_j)\| < L_0 \|\theta_i - \theta_j\| \quad (9)$$

$$\|Q'(x, \theta_i) - Q'(x, \theta_j)\| < L_1 \|\theta_i - \theta_j\| \quad (10)$$

In this case, it is easy to show that for any  $\theta$ , the mosaic risk function can provide a bounded approximation for the



empirical risk function:

$$\| R_N(\theta) - \tilde{R}_\epsilon(\theta) \| < \epsilon \cdot L_0 \quad (\forall \theta). \quad (11)$$

When we deal with a non-convex loss function, the mosaic risk function will give a very important benefit due to its low resolution: because we use a smaller codebook to approximate the training set, and one codeword may represent a large number of different training samples, the mosaic risk function normally corresponds to a smoother curve that may get rid of a lot of critical points comparing with the original empirical risk function. (Bengio et al. (2009) may support this argument.) Therefore, if we use the gradient descent method to optimize the mosaic risk function, named as *mosaic gradient descent (MGD)*, we can find its local minimum much easier and much faster, and this local minimum on mosaic risk function is a good initialization for further learning. If we can use a batch algorithm to optimize the mosaic risk function, it may significantly speed up the training phase due to a smaller number of codewords.

When we use MGD to minimize the mosaic risk function, we can get the following parameter update sequence:

$$\tilde{\theta}_{t+1} = \tilde{\theta}_t - \lambda_t \cdot \nabla_{\theta} \tilde{R}_\epsilon = \tilde{\theta}_t - \lambda_t \sum_{m=1}^M \frac{\omega_m}{N} \cdot \frac{\partial Q(c_m, \tilde{\theta}_t)}{\partial \theta} \quad (12)$$

Obviously, MGD generates a different sequence of the model parameters  $\tilde{\theta}_t$ .

Moreover, we may extend the above MGD to a stochastic version using only a random mini-batch of data for each model update in eq.(12) rather than the whole training set. All data in the selected mini-batch are approximated by codewords as in eq.(5). This is called mini-batch MGD. Of course, it may be better to use a much larger batch size in mini-batch MGD than that of mini-batch SGD to explore the overall structure of the mosaic function.

In the following, we will show that under some minor conditions, minimization of the mosaic risk function leads to convergence into a bounded neighborhood of a local optimum of the empirical risk function. Moreover, we also show that MGD may provide faster a convergence rate than GD and SGD under certain conditions.

### 3.1 CONVERGENCE ANALYSIS

As we know, if the learning rates satisfy some minor conditions, the batch GD algorithm in eq. (3) is guaranteed to converge to a critical point of the empirical risk function. In the following, let's first compare the MGD update sequence in eq.(12) with the GD update in eq.(3). Obviously, we have the following lemma:

**Lemma 1 (MGD vs. GD)** *Assume that the two update sequences in eq. (3) and eq. (12) start from the same initial*

*parameters  $\theta_0$ , and use the same sequence of learning rates  $\lambda_t$ , then we have:*

$$\| \tilde{\theta}_t - \hat{\theta}_t \| < \epsilon \cdot L_1 \cdot \sum_{\tau=1}^{t-1} \lambda_\tau \quad (13)$$

**Proof:** (1) At  $t = 1$ , assume that GD and MGD start from the same initialization  $\theta_0$  and share the same sequence of learning rate. Based on the Lipschitz-continuous condition in eq. (8), it is easy to show:

$$\begin{aligned} \| \tilde{\theta}_1 - \hat{\theta}_1 \| &= \frac{\lambda_0}{N} \left\| \sum_{n=1}^N \frac{\partial Q(x_n, \theta_0)}{\partial \theta_0} - \sum_{n=1}^N \frac{\partial Q(c^{(n)}, \theta_0)}{\partial \theta_0} \right\| \\ &\leq \frac{\lambda_0}{N} \cdot \sum_{n=1}^N \left\| \frac{\partial Q(x_n, \theta_0)}{\partial \theta_0} - \frac{\partial Q(c^{(n)}, \theta_0)}{\partial \theta_0} \right\| \\ &\leq \frac{\lambda_0}{N} \cdot L_1 \cdot \sum_{n=1}^N \| x_n - c^{(n)} \| \\ &\leq \epsilon \cdot L_1 \cdot \lambda_0 \end{aligned} \quad (14)$$

(2) Assume that the Lemma 1 holds for  $t$ , i.e.,

$$\| \tilde{\theta}_t - \hat{\theta}_t \| < \epsilon \cdot L_1 \cdot \sum_{\tau=1}^{t-1} \lambda_\tau. \quad (15)$$

For  $t + 1$ , considering the condition in eq. (10), we have:

$$\begin{aligned} &\| \tilde{\theta}_{t+1} - \hat{\theta}_{t+1} \| \\ &= \| (\tilde{\theta}_t - \hat{\theta}_t) + \frac{\lambda_t}{N} \sum_{n=1}^N \left( \frac{\partial Q(x_n, \hat{\theta}_t)}{\partial \theta} - \frac{\partial Q(c^{(n)}, \tilde{\theta}_t)}{\partial \theta} \right) \| \\ &\leq \| \tilde{\theta}_t - \hat{\theta}_t \| + \frac{\lambda_t}{N} \sum_{n=1}^N \left\| \frac{\partial Q(x_n, \hat{\theta}_t)}{\partial \theta} - \frac{\partial Q(c^{(n)}, \tilde{\theta}_t)}{\partial \theta} \right\| \\ &\leq \epsilon \cdot L_1 \cdot \sum_{\tau=1}^{t-1} \lambda_\tau + \lambda_t \cdot L_1 \cdot \| x_n - c^{(n)} \| \\ &= \epsilon \cdot L_1 \cdot \sum_{\tau=1}^t \lambda_\tau \end{aligned} \quad (16)$$

Therefore, Lemma 1 also holds for  $t + 1$ . ■

Lemma 1 means that if we run both MGD and the batch GD algorithm for  $t$  iterations, the difference between two resultant model parameters is bounded and it is proportional to the maximum quantization error,  $\epsilon$ , in the mosaic function.

Based on Lemma 1, we have the following theorem:

**Theorem 2 (MGD vs. GD)** *When we use the empirical risk function eq. (2) to measure the two parameters  $\tilde{\theta}_t$  in eq. (12) and  $\hat{\theta}_t$  in eq.(3), the difference is also bounded as:*

$$\| R_N(\tilde{\theta}_t) - R_N(\hat{\theta}_t) \| \leq \epsilon \cdot L_0 \cdot L_1 \cdot \sum_{\tau=1}^t \lambda_\tau \quad (17)$$

**Proof:** Based on the condition in eq.(7) we have:

$$\begin{aligned}
& \| R_N(\tilde{\theta}_t) - R_N(\hat{\theta}_t) \| = \\
& \frac{1}{N} \cdot \left\| \sum_{n=1}^N (Q(x_n, \tilde{\theta}_t) - Q(x_n, \hat{\theta}_t)) \right\| \\
& \leq \frac{1}{N} \cdot \sum_{n=1}^N \| Q(x_n, \tilde{\theta}_t) - Q(x_n, \hat{\theta}_t) \| \quad (18) \\
& \leq L_0 \cdot \| \tilde{\theta}_t - \hat{\theta}_t \| \\
& \leq L_0 \cdot \epsilon \cdot L_1 \cdot \sum_{\tau=1}^t \lambda_\tau. \quad \blacksquare
\end{aligned}$$

In Lemma 1 and Theorem 2, the bounds are proportional to the summation of all used learning rates. However, in many deep learning practices such as DNN/CNN training, we need to use a sequence of quickly-decayed learning rates to guarantee the convergence. In these situations, the summation of all learning rates is clearly bounded. Therefore, Theorem 2 shows that model parameters  $\tilde{\theta}_t$  derived by MGD provide a good estimation of  $\hat{\theta}_t$  learned by the regular batch GD algorithm when they are measured with the empirical risk function. The difference is bounded by a quantity proportional to the quantization error in the mosaic function. As a result, if the quantization error is sufficiently small, MGD converges into a bounded neighborhood of a critical point of the original empirical risk function.

### 3.2 FASTER LEARNING

Here we study the learning speed of MGD. If we want to optimize the empirical risk function  $R_N(\theta)$  up to a given precision  $\rho$ , i.e.,  $\| \theta - \theta^* \| < \rho$ , by using batch gradient descent in eq. (3), it will take  $O(\log \frac{1}{\rho})$  iterations, and the complexity of each iteration is  $O(N)$ . Thus the overall complexity of the batch algorithm is  $O(N \log(\frac{1}{\rho}))$  (Bottou, 2012).

Alternatively, we can run the MGD algorithm on eq. (12) for  $t$  iterations, and based on Lemma 1 we have:

**Lemma 3** *If we run the MGD algorithm in eq. (12) for  $t$  iterations, the model parameters can reach the precision as:*

$$\| \tilde{\theta}_t - \theta^* \| < \rho + \epsilon \cdot L_1 \cdot \sum_{\tau=1}^t \lambda_\tau \quad (19)$$

and the overall computational complexity of MGD is  $O(M \cdot t)$ .

Based on Lemma 3, we can have Theorem 4 as below:

**Theorem 4 (MGD vs. GD)** *Assume there exists a codebook  $C$  containing  $M$  codewords, which can approximate the whole training set well enough, and  $M$  is sufficiently*

*small, i.e.*

$$\epsilon \ll \frac{\rho}{L_1 \cdot \sum_{\tau=1}^t \lambda_\tau} \quad \text{and} \quad M < \frac{N \cdot O(\log(\frac{1}{\rho}))}{t} \quad (20)$$

*then to reach the same optimization precision, optimizing the mosaic risk function using MGD requires less computing resources and yields faster convergence speed than the batch GD in eq. (3).*

Similar to Theorem 4, we also have Theorem 5 that compares the resource requirement between MGD and SGD:

**Theorem 5 (MGD vs. SGD)** *In SGD, we need to run  $O(\frac{1}{\rho})$  iterations to achieve the optimization precision  $\rho$  (Bottou, 1998). Similar to Theorem 4, if we find a codebook which satisfies the quantization error requirement and remains sufficiently small as follows:*

$$\epsilon \ll \frac{\rho}{L_1 \cdot \sum_{\tau=1}^t \lambda_\tau} \quad \text{and} \quad M < \frac{1}{t} \cdot O(\frac{1}{\rho}) \quad (21)$$

*then MGD will require less computation resource than SGD to achieve the optimization precision  $\rho$ .*

In the case of big data, i.e.,  $N$  is extremely large ( $N \gg M$ ), or in an early stage of optimization, when we only require a rough optimization precision, i.e.,  $\rho$  is allowed to be relatively large, such codebook may exist. In these cases, it may be beneficial to run MGD instead of GD or SGD since MGD has faster convergence speed than batch GD and SGD. Moreover, as opposed to pure serial computation in SGD, the gradient computation in each MGD iteration can be easily parallelized. Therefore, MGD may provide an even faster training speed if multiple computing units are available.

## 4 ANNEALED GRADIENT DESCENT

Theorems 4 and 5 imply that MGD may possibly converge faster than either GD or SGD but it remains unclear how to find a codebook that simultaneously satisfy both conditions in these theorems. Moreover, we may require different levels of optimization precision in various stages of a training process. For example, at the beginning, when all model parameters stay far away from any local optimal point, we may not need to calculate a very accurate gradient, i.e.,  $\rho$  is allowed to be relatively large at this time. On the other hand, as the parameters move towards a close neighborhood of an optimal point, we may require a very small  $\rho$  to perform an accurate local search to converge more effectively. As suggested by eq.(20), the required quantization error,  $\epsilon$ , is proportionally related to  $\rho$ . For a fixed set of training data, the quantization error,  $\epsilon$ , in turn depends on the size of the codebook,  $M$ . This suggests we use an annealing schedule of  $\{\epsilon_1, \epsilon_2, \dots\}$  (with  $\epsilon_{i+1} < \epsilon_i$ ) for

the whole training process, where  $\epsilon$  gradually decreases as training continues. At the beginning, we can use a small low resolution codebook (relatively big  $\epsilon$ ) to run MGD to learn model parameters. As training proceeds, we gradually reduce  $\epsilon$  by using increasingly larger codebooks. At the final stage, we may even use all original training samples to fine-tune the model parameters.

Therefore, the basic idea of annealed gradient descent (AGD) is to construct deeply-structured hierarchical codebooks, in which quantization error  $\epsilon$  slowly decreases from the top layer down to the bottom layer, and the last layer is finally connected to the original training set. During the training procedure, we first start from the top to use each layer of codebooks to do MGD updates in eq.(12) and gradually move down the hierarchy based on a pre-specified annealing schedule until we finally use the training samples to fine-tune the model parameters. If a proper annealing schedule is used, this annealed learning process may accelerate the training speed as implied by Theorems 4 and 5. More importantly, it may help to converge to a better local optimum at the end because AGD optimizes much smoother mosaic objective functions from the early stage of training.

In this section, we first briefly discuss the hierarchical codebooks, and then present the AGD training algorithm.

### 4.1 HIERARCHICAL CODEBOOKS

In this work, we use a regular K-means based top-down hierarchical clustering algorithm (Zhou et al., 2015) to construct the required hierarchical codebooks, where the centroid of each cluster is used as a codeword for each layer. The structure of the hierarchical codebooks is shown in Figure 1.

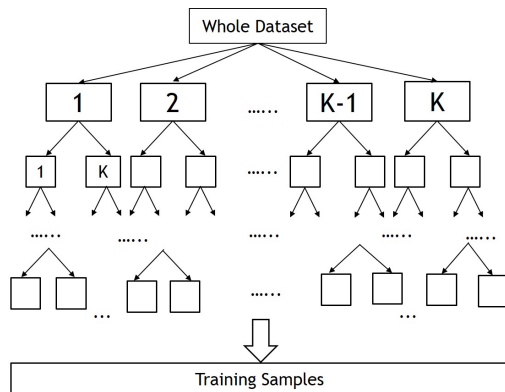


Figure 1: Illustration of a hierarchical codebook for AGD

When building the codebooks, we first divide the training set into several subsets based on the class labels of data samples (each subset only contains all training samples

from one class label), then conduct the hierarchical top-down K-means clustering on each subset. Note that the clustering process is very easy to parallelize because all subsets are independent with each other and we can run hierarchical K-means on them separately. In the K-means clustering, we first define a suitable  $K$ , then use K-means to split each subset into  $K$  clusters. The centroids of all clusters are used to build the first layer of codebooks. Next, we continuously apply the K-means clustering on all clusters to further divide them into  $K$  sub-clusters to derive the codebooks in the next layer. We repeat this procedure several times until the quantization error in the last layer becomes small enough. Finally, we connect the original training samples at the bottom as the leaf nodes to obtain a hierarchical codebook as shown in Figure 1, in which the sizes of the codebooks are gradually increased from the top layer to the bottom layer. In this way, the K-means codewords in the various layers of the hierarchy may be used to approximate each training sample in the leaf node up to different precision levels as required in the following AGD algorithm.

### 4.2 ANNEALED GRADIENT DESCENT ALGORITHM

In AGD, we first specify an annealing schedule, i.e.,  $\{\epsilon_1, \epsilon_2, \dots\}$  ( $\epsilon_{i+1} < \epsilon_i$ ). In each epoch of AGD, for each training sample in the selected data mini-batch, we select a codeword from the uppermost layer of the hierarchical codebooks that barely satisfies the required quantization error  $\epsilon_i$ , to construct the mosaic function for MGD at this stage. Since  $\epsilon_i$  gradually decreases in the annealing schedule, we slowly move to use more and more precise codewords (eventually the original data samples) in AGD. In AGD, a hierarchical search list is constructed for each layer in  $C$  based on the average quantization errors in K-means. As a result, in each AGD step, the corresponding codewords can be found very efficiently based on this search list. The annealed gradient descent (AGD) algorithm is shown as in Algorithm 1. During the AGD training, we may even use varying batch sizes. For example, we can start from a very large batch size (even the whole training set) at the beginning and slowly decrease the batch size from epoch to epoch. In general, we normally use much larger batch sizes than those used in the regular mini-batch SGD. If a suitable annealing schedule is specified, many initial AGD epochs may be designated to run faster MGD with smaller codebooks, yielding faster training speed than the mini-batch SGD or GD in overall.

## 5 EXPERIMENTS

In this section, we apply the proposed AGD algorithm to learning sigmoid fully connected deep neural networks (DNNs) for image recognition in the MNIST database and

---

**Algorithm 1** Annealed Gradient Descent(AGD)

---

**Input:** training set  $O$ , hierarchical codebook  $C$ , annealing schedule  $\{\epsilon_1, \epsilon_2, \dots, \epsilon_T \mid \epsilon_{i+1} < \epsilon_i\}$   
**for each** epoch  $i = 1$  **to**  $T$  **do**  
  **for each** batch  $X$  **do**  
    For each sample in  $X$ , select a codeword  $c^{(n)}$  at the uppermost layer of the  $C$  satisfying  $\epsilon_i$ ;  
    Use MGD to optimize the mosaic risk function;  
  **end for**  
**end for**

---

Table 1: The total training time of K-means clustering.

| Database    | Number of CPUs | Training Time |
|-------------|----------------|---------------|
| MNIST       | 5              | 4.3 (hr)      |
| Switchboard | 8              | 9.2 (hr)      |

speech recognition in the Switchboard database. AGD and the regular mini-batch SGD are both used to train DNNs based on the minimum cross-entropy error criterion. AGD is compared with mini-batch SGD in terms of the total training time and the final recognition performance.

For each data set, we first use a standard K-means algorithm to build a deeply-structured hierarchical codebook. We use  $K = 5$  in MNIST and  $K = 4$  in Switchboard, which can result in a hierarchical codebook with sufficient depth. In our experiments, the MNIST database contains 10 classes and Switchboard contains 8991 classes, thus the hierarchical K-means has a good potential for parallel training. Here, the K-means clustering procedure is parallelized among multiple CPUs to speed it up as much as possible. In our experiments, the total running time of the K-means clustering is about 4.3 hours in MNIST (using 5 CPUs) and about 9.2 hours in Switchboard (using 8 CPUs), as shown in Table 1. If we use more CPUs, the K-means running time can be further reduced. As shown later, the K-means training time is not significant when comparing with the necessary DNN training times. And if we use more CPUs to do K-means, the clustering time can be further reduced (approximately less than 3 hours for MNIST and less than 5 hours for Switchboard). Moreover, for each database we only need to run K-means once and after that we can use the same codebook to train DNNs based on different annealing schedules. Therefore, we do not take into account the running time of the K-means clustering in the following comparisons. Note that no pre-training is used for DNNs in our experiments.

## 5.1 MNIST: IMAGE RECOGNITION

The MNIST database (LeCun et al., 1998) contains 60,000 training images and 10,000 test images, and the images are 28-by-28 in size. Here, we use data augmentation through image deformation at the beginning of each epoch to en-

large the training set as in (Ciresan et al., 2010). We use the configuration of 3-hidden-layer DNNs (1500, 1000, 500 nodes in each hidden layer) in (Ciresan et al., 2010) as our network structures and use SGD and AGD to do network training. Following (Ciresan et al., 2010), we fine-tune all hyper-parameters towards the best possible performance. In SGD, we use a mini-batch of 10 samples and an initial learning rate of 0.4. Notice that in MNIST experiments we do not use momentum during the training phase. In MGD, we use a larger mini-batch size of 4500 and an initial learning rate of 0.8. The training process runs for 550 epochs to guarantee the convergence of the augmented MNIST database.

As we know, we should shrink the learning rates during the training process for better convergence. Specifically, when the training mean square error (MSE) becomes smaller than a pre-defined threshold  $r$ , we use the formula

$$\lambda_{t+1} = \lambda_t \cdot \frac{p}{p+t}$$

to gradually decrease the learning rate, where  $p$  is a pre-defined constant to control the decreasing speed of the learning rates. In our experiments, we set  $r = 0.17$  and  $p = 10000$ . As for the AGD annealing schedule in MNIST, we start from  $\epsilon_1 = 7.5$  (this value is based on the average quantization error in the first layer of the codebook) and  $\epsilon_{i+1} = 0.999 \cdot \epsilon_i$ .

During the annealing phase we use MGD to train the network while in the regular phase we use SGD with the same configurations as the baseline. Note that we only do image deformation during the regular phase. In Figure 2, we have shown the learning curves of both SGD and AGD in terms of cross-entropy and classification error rate on the MNIST training set. Since each MGD epoch runs much faster than a SGD epoch, all learning curves are plotted as a function of total training time instead of epochs.

From the two pictures in Figure 2 we can see that AGD (in blue) finishes the same number of epochs much earlier than SGD (in red). In addition, in Table 2, we also give the total training time and the best classification error on the test set for both AGD and SGD. From results in Figure 2 and Table 2, we can see that the proposed AGD training algorithm yields slightly better classification performance in the test set, and more importantly reduces the total DNN training time by about 40%.

## 5.2 SWITCHBOARD: SPEECH RECOGNITION

Switchboard is a 320-hour English transcription task, which contains 332,576 utterances in its training set (amounting to about 126 millions of training samples in total). We select the standard NIST Hub5e2000 as the test

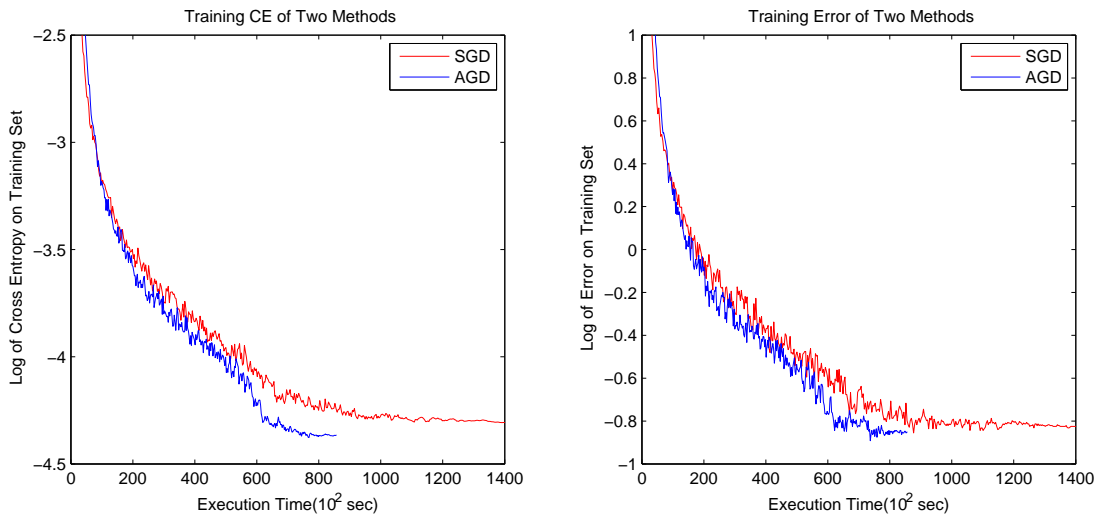


Figure 2: Learning curves on the MNIST training set (Left: cross entropy error; Right: classification error) of SGD and AGD as a function of elapsed training time.

Table 2: Comparison between SGD and AGD in terms of total training time (using one GPU) and the best classification error rate on MNIST.

| Method | Training Time | Test Error |
|--------|---------------|------------|
| SGD    | 38.8 (hr)     | 0.47%      |
| AGD    | 23.6 (hr)     | 0.46%      |

set in this work, which has 1831 utterances.<sup>1</sup> Following (Seide et al., 2011; Bao et al., 2013; Pan et al., 2012), we train a 6-hidden-layer DNN with 2048 nodes per layer based on the minimum cross-entropy criterion. We compare the cross entropy and frame classification errors on the training and test sets to evaluate the performance of SGD and AGD, meanwhile we also evaluate word error rates in speech recognition for the test set.

Here we use similar hyper-parameters as in (Pan et al., 2012; Xue et al., 2014). For example, we use a mini-batch of 1024 samples and an initial learning rate of 0.2 in SGD, and a mini-batch of 6144 and an initial learning rate of 1.0 in MGD. We use 0.9 as the momentum in both SGD and MGD. We run 10 epochs in SGD and 17 epochs in AGD. During the training process, we need to shrink the learning rates slowly. Specifically, we multiply the learning rate by 0.8 every epoch after the 5-th epoch in SGD and the 12-th epoch in AGD. Note that our SGD baseline is solid and comparable with the best results reported on this task

<sup>1</sup>Due to the copyright issue, all Switchboard experiments were conducted at NELSLIP, University of Science and Technology of China.

(Seide et al., 2011; Hinton et al., 2012; Xue et al., 2014) in terms of both training speed and recognition performance.

As for the AGD annealing schedule in Switchboard, unlike MNIST, it starts from an initial value  $\epsilon_1$  (17.5 in this case), and use the formula

$$\epsilon_{i+1} = \epsilon_i - \Delta_\epsilon$$

to reduce  $\epsilon_i$  by subtracting a constant value every epoch until it reaches the pre-defined value (8.5 in this case). The reason for this formula is that we run much less epochs here. In Switchboard, we have evaluated three different annealing schedules to show how they affect the training speed and the final recognition performance as shown in Table 3. In these three schedules, we use different values for  $\Delta_\epsilon$ , e.g. 1.0, 0.9 and 0.8 respectively. We can see that the annealing schedule 1 ( $\Delta_\epsilon = 1.0$ ) decreases fastest and it provides the best performance but relatively slower training speed. In contrast, schedule 3 ( $\Delta_\epsilon = 0.8$ ) gives the fastest training speed but slightly worse performance because more epochs will be dispatched to run MGD.

Figure 3 shows the learning curves of both SGD and AGD (using the annealing schedule with  $\Delta_\epsilon = 1$ ) in terms of cross-entropy and frame errors on the Switchboard training set as a function of elapsed training time. Clearly, AGD runs much faster than SGD on Switchboard as well. Meanwhile, AGD can also achieve a slightly better local minimum than SGD as measured in both figures.

In Table 3, we give the total training times and the word error rates in speech recognition. We report the experimental results for all 3 different annealing schedules. The results have shown that the proposed AGD method can yield similar recognition performance with much faster training

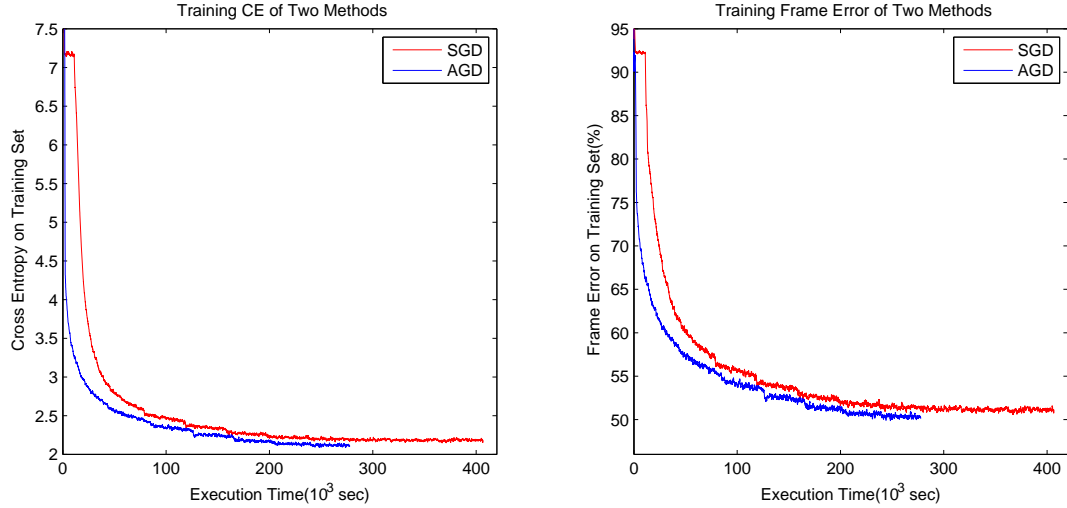


Figure 3: Learning curves on the Switchboard training set (Left: cross entropy error; Right: frame classification error) of SGD and AGD as a function of elapsed training time.

Table 3: Comparison between SGD and AGD in terms of total training time (using one GPU) and word error rate in speech recognition on Switchboard.

| Method                          | Training Time | Word Error |
|---------------------------------|---------------|------------|
| SGD                             | 114.05 (hr)   | 16.4%      |
| AGD ( $\Delta_\epsilon = 1.0$ ) | 78.79 (hr)    | 16.3%      |
| AGD ( $\Delta_\epsilon = 0.9$ ) | 66.63 (hr)    | 16.7%      |
| AGD ( $\Delta_\epsilon = 0.8$ ) | 55.35 (hr)    | 17.5%      |

speed than SGD (about a 30% to 40% reduction in total training time) when a suitable annealing schedule is used. Results in Table 3 also imply how quantization errors in the codebooks may affect the final classification performance: a slower schedule means more epochs will be dispatched to run MGD, which uses each layer of the codebooks to train the DNNs. In this case, the quantization error of the codebook may bring about some negative influence on the performance but provide faster learning speed. In practice, when we are sensitive to the total training time of DNNs, we may use slower decreasing schedules to accelerate the DNN training dramatically.

## 6 CONCLUSIONS

In this paper, we have proposed a new annealed gradient descent (AGD) algorithm for non-convex optimization in deep learning, which can converge to a better local minimum with faster speed when compared with the regular mini-batch SGD algorithm. In this work, AGD has been applied to training large scale DNNs for image classifi-

cation and speech recognition tasks. Experimental results have shown that AGD significantly outperforms SGD in terms of the convergence speed. Therefore, the AGD algorithm is especially suitable for the large scale non-convex optimization problems in deep learning. In the future, we may apply AGD to training convolutional neural networks (CNNs) for other more challenging image recognition tasks, where we may build the hierarchical codebooks by clustering image patches instead of the whole input images. Moreover, AGD may be applied to the recently proposed unsupervised learning algorithm in (Zhang and Jiang, 2015; Zhang et al., 2015) as well.

## Acknowledgments

This work was partially supported by an NSERC discovery grant from the Canadian Federal Government. The first author is supported by a scholarship from China Scholarship Council (CSC). We appreciate Dr. Pan Zhou from NELSIP, University of Science and Technology of China for his help in conducting the Switchboard experiments in this paper.

## References

- A. Agarwal and J. C. Duchi. Distributed delayed stochastic optimization. In *IEEE Annual Conference on Decision and Control (CDC)*, pages 5451–5452, 2012.
- Y. Bao, H. Jiang, L. Dai, and C. Liu. Incoherent training of deep neural networks to de-correlate bottleneck features for speech recognition. In *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 6980–6984, 2013.

- Y. Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009.
- Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48. ACM, 2009.
- C. Bockermann and S. Lee. Scalable stochastic gradient descent with improved confidence. In *NIPS Workshop on Big Learning—Algorithms, Systems, and Tools for Learning at Scale*, 2012.
- L. Bottou. Online learning and stochastic approximations. *On-line learning in neural networks*, 17:9, 1998.
- L. Bottou. Stochastic learning. In *Advanced lectures on machine learning*, pages 146–168. Springer, 2004.
- L. Bottou. Stochastic gradient tricks. *Neural networks: tricks of the trade*. Springer, Berlin, pages 430–445, 2012.
- A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, and Y. LeCun. The loss surface of multilayer networks. *arXiv preprint arXiv:1412.0233*, 2014.
- D. C. Ciresan, U. Meier, L. M. Gambardella, and J. Schmidhuber. Deep big simple neural nets excel on handwritten digit recognition. *Neural Computation*, 22(12):3207–3220, 2010.
- Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in Neural Information Processing Systems (NIPS’14)*, pages 2933–2941, 2014.
- J. L. Elman. Learning and development in neural networks: The importance of starting small. *Cognition*, 48(1):71–99, 1993.
- N. Feng, R. Benjamin, R. Christopher, and J. W. Stephen. Hogwild!: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in Neural Information Processing System 24 (NIPS’11)*, 2011.
- G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury. Deep neural networks for acoustic modelling in speech recognition. *IEEE Signal Processing Magazine*, 29, 2012.
- K. A. Krueger and P. Dayan. Flexible shaping: How learning in small steps helps. *Cognition*, 110(3):380–394, 2009.
- Y. LeCun and L. Bottou. Large scale online learning. In *Advances in neural information processing systems 17 (NIPS’04)*, page 217, 2004.
- Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Back-propagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.
- Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- N. Murata and S. Amari. Statistical analysis of learning dynamics. *Signal Processing*, 74(1):3–28, 1999.
- J. Pan, C. Liu, Z. Wang, Y. Hu, and H. Jiang. Investigations of deep neural networks for large vocabulary continuous speech recognition: Why DNN surpasses GMMs in acoustic modelling. In *Proc. of International Symposium on Chinese Spoken Language Processing*, 2012.
- N. L. Roux, M. Schmidt, and F. R. Bach. A stochastic gradient method with an exponential convergence rate for finite training sets. In *Advances in Neural Information Processing Systems 25 (NIPS’12)*, pages 2672–2680, 2012.
- F. Seide, G. Li, and D. Yu. Conversational speech transcription using context-dependent deep neural networks. In *Proc. of Interspeech*, pages 437–440, 2011.
- O. Shamir and T. Zhang. Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 71–79, 2013.
- V. N. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, 1st edition, 1998.
- S. Xue, O. Abdel-Hamid, H. Jiang, L. Dai, and Q. Liu. Fast adaptation of deep neural network based on discriminant codes for speech recognition. *IEEE/ACM Trans. on Audio, Speech and Language Processing*, 22(12):1713–1725, 2014.
- S. Zhang and H. Jiang. Hybrid orthogonal projection and estimation (hope): A new framework to probe and learn neural networks. In *arXiv:1502.00702*, 2015.
- S. Zhang, L. Dai, and H. Jiang. The new hope way to learn neural networks. In *Proc. of Deep Learning Workshop at ICML 2015*, 2015.
- P. Zhou, H. Jiang, L. Dai, Y. Hu, and Q. Liu. State-clustering based multiple deep neural networks modeling approach for speech recognition. *IEEE/ACM Trans. on Audio, Speech and Language Processing*, 23(4):631–642, 2015.
- M. Zinkevich, J. Langford, and E. J. Smola. Slow learners are fast. In *Advances in Neural Information Processing Systems 22 (NIPS’09)*, pages 2331–2339, 2009.

---

# Max-Product Belief Propagation for Linear Programming: Applications to Combinatorial Optimization

---

**Sejun Park**

Department of Electrical Engineering  
Korea Advanced Institute of Science and Technology  
sejun.park@kaist.ac.kr

**Jinwoo Shin**

Department of Electrical Engineering  
Korea Advanced Institute of Science and Technology  
jinwoos@kaist.ac.kr

## Abstract

Max-product belief propagation (BP) is a popular message-passing algorithm for computing a maximum-a-posteriori (MAP) assignment in a joint distribution represented by a graphical model (GM). It has been shown that BP can solve a few classes of Linear Programming (LP) formulations to combinatorial optimization problems including maximum weight matching and shortest path, i.e., BP can be a distributed solver for certain LPs. However, those LPs and corresponding BP analysis are very sensitive to underlying problem setups, and it has been not clear what extent these results can be generalized to. In this paper, we obtain a generic criteria that BP converges to the optimal solution of given LP, and show that it is satisfied in LP formulations associated to many classical combinatorial optimization problems including maximum weight perfect matching, shortest path, traveling salesman, cycle packing and vertex cover. More importantly, our criteria can guide the BP design to compute fractional LP solutions, while most prior results focus on integral ones. Our results provide new tools on BP analysis and new directions on efficient solvers for large-scale LPs.

## 1 INTRODUCTION

Graphical model (GM) has been one of powerful paradigms for succinct representations of joint probability distributions in variety of scientific fields (Yedidia et al., 2005; Richardson and Urbanke, 2008; Mezard and Montanari, 2009; Wainwright and Jordan, 2008). GM represents a joint distribution of some random vector to a graph structured model where each vertex corresponds to a random variable and each edge captures to a conditional dependency between random variables. In many applications involving GMs, finding maximum-a-posteriori (MAP) assignment in GM is an important inference task, which is

known to be computationally intractable (i.e., NP-hard) in general (Chandrasekaran et al., 2008). Max-product belief propagation (BP) is the most popular heuristic for approximating a MAP assignment of given GM, where its performance has been not well understood in loopy GMs. Nevertheless, BP often shows remarkable performances even on loopy GM. Distributed implementation, associated ease of programming and strong parallelization potential are the main reasons for the growing popularity of the BP algorithm. For example, several software architectures for implementing parallel BPs were recently proposed (Low et al., 2010; Gonzalez et al., 2010; Ma et al., 2012) by different research groups in machine learning communities.

In the past years, there have been made extensive research efforts to understand BP performances on loopy GMs behind its empirical success. Several characterizations of the max-product BP fixed points have been proposed (Weiss and Freeman, 2001; Vinyals et al., 2010), whereas they do not guarantee the BP convergence in general. It has also been studied about the BP convergence to the correct answer, in particular, under a few classes of loopy GM formulations of combinatorial optimization problems: matching (Bayati et al., 2005; Sanghavi et al., 2011; Huang and Jebara, 2007; Salez and Shah, 2009), perfect matching (Bayati et al., 2011), matching with odd cycles (Shin et al., 2013) and shortest path (Ruozzi and Tatikonda, 2008). The important common feature of these instances is that BP converges to a correct MAP assignment if the Linear Programming (LP) relaxation of the MAP inference problem is tight, i.e., it has no integrality gap. In other words, BP can be used an efficient distributed solver for those LPs, and is presumably of better choice than classical centralized LP solvers such as simplex methods (Dantzig, 1998), interior point methods (Thapa, 2003) and ellipsoid methods (Khachiyan, 1980) for large-scale inputs. However, these theoretical results on BP are very sensitive to underlying structural properties depending on specific problems and it is not clear what extent they can be generalized to, e.g., the BP analysis for matching problems (Bayati et al., 2005; Sanghavi et al., 2011; Huang and Jebara, 2007; Salez and Shah, 2009) are not extended to even for perfect matching



ones (Bayati et al., 2011). In this paper, we overcome such technical difficulties for enhancing the power of BP as a LP solver.

**Contribution.** We establish a generic criteria for GM formulations of given LP so that BP converges to the optimal LP solution. By product, it also provides a sufficient condition for a unique BP fixed point. As one can naturally expect given prior results, one of our conditions requires the LP tightness. Our main contribution is finding other sufficient generic conditions so that BP converges to the correct MAP assignment of GM. First of all, our generic criteria can rediscover all prior BP results on this line, including matching (Bayati et al., 2005; Sanghavi et al., 2011; Huang and Jebara, 2007), perfect matching (Bayati et al., 2011), matching with odd cycles (Shin et al., 2013) and shortest path (Ruozi and Tatikonda, 2008), i.e., we provide a unified framework on establishing the convergence and correctness of BPs in relation to associated LPs. Furthermore, we provide new instances under our framework: we show that BP can solve LP formulations associated to other popular combinatorial optimizations including perfect matching with odd cycles, traveling salesman, cycle packing and vertex cover, which are not known in the literature. While most prior known BP results on this line focused on the case when the associated LP has an integral solution, the proposed criteria naturally guides the BP design to compute fractional LP solutions as well (see Section 4.2 and Section 4.4 for details).

Our proof technique is built upon on that of Sanghavi et al. (2011) where the authors construct an alternating path in the computational tree induced by BP to analyze its performance for the maximum weight matching problem. Such a trick needs specialized case studies depending on the associated LP when the path reaches a leaf of the tree, and this is one of main reasons why it is not easy to generalize to other problems beyond matching. The main technical contribution of this paper is providing a way to avoid the issue in the BP analysis via carefully analyzing associated LP polytopes.

The main appeals of our results are providing not only tools on BP analysis, but also guidelines on BP design for its high performance, i.e., one can carefully design a BP given LP so that it satisfies the proposed criteria. We run such a BP for solving the famous traveling salesman problem (TSP), and our experiments show that BP outperforms other popular heuristics (see Section 5). Our results provide not only new tools on BP analysis and design, but also new directions on efficient distributed (and parallel) solvers for large-scale LPs and combinatorial optimization problems.

**Organization.** In Section 2, we introduce necessary backgrounds for the BP algorithm. In Section 3, we provide the main result of the paper, and several concrete applications to popular combinatorial optimizations are described

in Section 4. In Section 5, we show empirical performances of BP algorithms for solving TSP.

## 2 PRELIMINARIES

### 2.1 GRAPHICAL MODEL

A joint distribution of  $n$  (binary) random variables  $Z = [Z_i] \in \{0, 1\}^n$  is called a Graphical Model (GM) if it factorizes as follows: for  $z = [z_i] \in \{0, 1\}^n$ ,

$$\Pr[Z = z] \propto \prod_{i \in \{1, \dots, n\}} \psi_i(z_i) \prod_{\alpha \in F} \psi_\alpha(z_\alpha),$$

where  $\{\psi_i, \psi_\alpha\}$  are (given) non-negative functions, so-called factors;  $F$  is a collection of subsets

$$F = \{\alpha_1, \alpha_2, \dots, \alpha_k\} \subset 2^{\{1, 2, \dots, n\}}$$

(each  $\alpha_j$  is a subset of  $\{1, 2, \dots, n\}$  with  $|\alpha_j| \geq 2$ );  $z_\alpha$  is the projection of  $z$  onto dimensions included in  $\alpha$ .<sup>1</sup> In particular,  $\psi_i$  is called a variable factor. Figure 1 depicts the graphical relation between factors  $F$  and variables  $z$ .

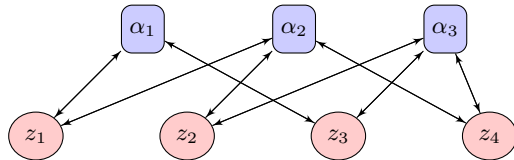


Figure 1: Factor graph for the graphical model  $\Pr[z] \propto \psi_{\alpha_1}(z_1, z_3)\psi_{\alpha_2}(z_1, z_2, z_3)\psi_{\alpha_3}(z_2, z_3, z_4)$ , i.e.,  $F = \{\alpha_1, \alpha_2, \alpha_3\}$  and  $n = 4$ . Each  $\alpha_j$  selects a subset of  $z$ . For example,  $\alpha_1$  selects  $\{z_1, z_3\}$ .

Assignment  $z^*$  is called a maximum-a-posteriori (MAP) assignment if  $z^* = \arg \max_{z \in \{0, 1\}^n} \Pr[z]$ . This means that computing a MAP assignment requires us to compare  $\Pr[z]$  for all possible  $z$ , which is typically computationally intractable (i.e., NP-hard) unless the induced bipartite graph of factors  $F$  and variables  $z$ , so-called factor graph, has a bounded treewidth (Chandrasekaran et al., 2008).

### 2.2 MAX-PRODUCT BELIEF PROPAGATION

The (max-product) BP algorithm is a popular heuristic for approximating the MAP assignment in GM. BP is implemented iteratively; at each iteration  $t$ , BP maintains four messages  $\{m_{\alpha \rightarrow i}^t(c), m_{i \rightarrow \alpha}^t(c) : c \in \{0, 1\}\}$  between every variable  $z_i$  and every associated  $\alpha \in F_i$ , where  $F_i := \{\alpha \in F : i \in \alpha\}$ ; that is,  $F_i$  is a subset of  $F$  such that all  $\alpha$  in  $F_i$  are associated with  $z_i$ . The messages

<sup>1</sup>For example, if  $z = [0, 1, 0]$  and  $\alpha = \{1, 3\}$ , then  $z_\alpha = [0, 0]$ .

are updated as follows:

$$m_{\alpha \rightarrow i}^{t+1}(c) = \max_{z_\alpha: z_i=c} \psi_\alpha(z_\alpha) \prod_{j \in \alpha \setminus i} m_{j \rightarrow \alpha}^t(z_j) \quad (1)$$

$$m_{i \rightarrow \alpha}^{t+1}(c) = \psi_i(c) \prod_{\alpha' \in F_i \setminus \alpha} m_{\alpha' \rightarrow i}^t(c). \quad (2)$$

Where each  $z_i$  only sends messages to  $F_i$ ; that is,  $z_i$  sends messages to  $\alpha_j$  only if  $\alpha_j$  selects/includes  $i$ . The outer-term in the message computation (1) is maximized over all possible  $z_\alpha \in \{0, 1\}^{|\alpha|}$  with  $z_i = c$ . The inner-term is a product that only depends on the variables  $z_j$  (excluding  $z_i$ ) that are connected to  $\alpha$ . The message-update (2) from variable  $z_i$  to factor  $\psi_\alpha$  is a product containing all messages received by  $z_i$  in the previous iteration, except for the message sent by  $\psi_\alpha$  itself.

One can reduce the complexity by combining (1) and (2) as:

$$m_{i \rightarrow \alpha}^{t+1}(c) = \psi_i(c) \prod_{\alpha' \in F_i \setminus \alpha} \max_{z_{\alpha'}: z_i=c} \psi_{\alpha'}(z_{\alpha'}) \times \prod_{j \in \alpha' \setminus i} m_{j \rightarrow \alpha'}^t(z_j).$$

The BP fixed-point of messages is defined as  $m^{t+1} = m^t$  under the above updating rule. Given a set of messages  $\{m_{i \rightarrow \alpha}(c), m_{\alpha \rightarrow i}(c) : c \in \{0, 1\}\}$ , the so-called BP marginal beliefs are computed as follows:

$$b_i[z_i] = \psi_i(z_i) \prod_{\alpha \in F_i} m_{\alpha \rightarrow i}(z_i). \quad (3)$$

This BP algorithm outputs  $z^{BP} = [z_i^{BP}]$  where

$$z_i^{BP} = \begin{cases} 1 & \text{if } b_i[1] > b_i[0] \\ ? & \text{if } b_i[1] = b_i[0] \\ 0 & \text{if } b_i[1] < b_i[0] \end{cases}.$$

It is known that  $z^{BP}$  converges to a MAP assignment after a sufficient number of iterations, if the factor graph is a tree and the MAP assignment is unique. However, if the graph contains cycles, the BP algorithm is not guaranteed to converge a MAP assignment in general.

### 3 CONVERGENCE AND CORRECTNESS OF BELIEF PROPAGATION

In this section, we provide the main result of this paper: a convergence and correctness criteria of BP. Consider the following GM: for  $x = [x_i] \in \{0, 1\}^n$  and  $w = [w_i] \in \mathbb{R}^n$ ,

$$\Pr[X = x] \propto \prod_i e^{-w_i x_i} \prod_{\alpha \in F} \psi_\alpha(x_\alpha), \quad (4)$$

where  $F$  is the set of non-variable factors and the factor function  $\psi_\alpha$  for  $\alpha \in F$  is defined as

$$\psi_\alpha(x_\alpha) = \begin{cases} 1 & \text{if } A_\alpha x_\alpha \geq b_\alpha, C_\alpha x_\alpha = d_\alpha \\ 0 & \text{otherwise} \end{cases},$$

for some matrices  $A_\alpha, C_\alpha$  and vectors  $b_\alpha, d_\alpha$ . Now we consider the Linear Programming (LP) corresponding the above GM:

$$\begin{aligned} & \text{minimize} && w \cdot x \\ & \text{subject to} && \psi_\alpha(x_\alpha) = 1, \quad \forall \alpha \in F \\ & && x = [x_i] \in [0, 1]^n. \end{aligned} \quad (5)$$

One can easily observe that the MAP assignments for GM (4) corresponds to the (optimal) solution of LP (5) if the LP has an integral solution  $x^* \in \{0, 1\}^n$ . As stated in the following theorem, we establish other sufficient conditions so that the max-product BP can indeed find the LP solution.

**Theorem 1** *The max-product BP on GM (4) with arbitrary initial message converges to the solution of LP (5) if the following conditions hold:*

- C1. LP (5) has a unique integral solution  $x^* \in \{0, 1\}^n$ , i.e., it is tight.
- C2. For every  $i \in \{1, 2, \dots, n\}$ , the number of factors associated with  $x_i$  is at most two, i.e.,  $|F_i| \leq 2$ .
- C3. For every factor  $\psi_\alpha$ , every  $x_\alpha \in \{0, 1\}^{|\alpha|}$  with  $\psi_\alpha(x_\alpha) = 1$ , and every  $i \in \alpha$  with  $x_i \neq x_i^*$ , there exists  $\gamma \subset \alpha$  such that

$$|\{j \in \{i\} \cup \gamma : |F_j| = 2\}| \leq 2$$

$$\begin{aligned} \psi_\alpha(x'_\alpha) = 1, & \quad \text{where } x'_k = \begin{cases} x_k & \text{if } k \notin \{i\} \cup \gamma \\ x_k^* & \text{otherwise} \end{cases} \\ \psi_\alpha(x''_\alpha) = 1, & \quad \text{where } x''_k = \begin{cases} x_k & \text{if } k \in \{i\} \cup \gamma \\ x_k^* & \text{otherwise} \end{cases} \end{aligned}.$$

Since Theorem 1 holds for arbitrary initial messages, the conditions C1, C2, C3 also provides the uniqueness of BP fixed-points in term of marginal beliefs, as follows.

**Corollary 2** *The BP fixed-points of GM (4) have the same marginal beliefs if conditions C1, C2, C3 hold.*

The conditions C2, C3 are typically easy to check given GM (4) and the uniqueness in C1 can be easily guaranteed via adding random noises, where we provide several concrete examples in Section 4. On the other hand, the integral property in C1 requires to analyze LP (5), where it has been extensively studied in the field of combinatorial optimization (Schrijver, 2003). Nevertheless, Theorem 1 provides important guidelines to design BP algorithms, ir-respectively of the LP analysis. For example, in Section 5, we report empirical performances of BP following the above guideline for solving the traveling salesman problem, without relying on whether the corresponding LP has an integral solution or not.

### 3.1 PROOF OF THEOREM 1

To begin with, we define some necessary notation. We let  $\mathcal{P}$  denote the polytope of feasible solutions of LP (5):

$$\mathcal{P} := \{x \in [0, 1]^n : \psi_\alpha(x_\alpha) = 1, \forall \alpha \in F\}.$$

Similarly,  $\mathcal{P}_\alpha$  is defined as

$$\mathcal{P}_\alpha := \left\{x \in [0, 1]^{|\alpha|} : \psi_\alpha(x_\alpha) = 1\right\}.$$

We first state the following key technical lemma.

**Lemma 3** *There exist universal constants  $K, \eta > 0$  for LP (5) such that if  $z \in [0, 1]^n$  and  $0 < \varepsilon < \eta$  satisfy the followings:*

1. *There exist at most two violated factors for  $z$ , i.e.,  $|\{\alpha \in F : z_\alpha \notin \mathcal{P}_\alpha\}| \leq 2$ .*
2. *For each violated factor  $\alpha$ , there exist  $i \in \alpha$  such that  $z_\alpha^\dagger \in \mathcal{P}_\alpha$ , where  $z^\dagger = z + \varepsilon e_i$  or  $z^\dagger = z - \varepsilon e_i$  and  $e_i \in \{0, 1\}^n$  is the unit vector whose  $i$ -th coordinate is 1,*

*then there exists  $z^\ddagger \in \mathcal{P}$  such that  $\|z - z^\ddagger\|_1 \leq \varepsilon K$ .*

The proof of Lemma 3 is presented in Section 3.2. Now, from Condition C1, it follows that there exists  $\rho > 0$  such that

$$\rho := \inf_{x \in \mathcal{P} \setminus x^*} \frac{w \cdot x - w \cdot x^*}{\|x - x^*\|_1} > 0. \quad (6)$$

We let  $\hat{x}^t \in \{0, 1, ?\}^n$  denote the BP estimate at the  $t$ -th iteration for the MAP computation. We will show that under Conditions C1-C3,

$$\hat{x}^t = x^*, \quad \text{for } t > \left(\frac{w_{\max}}{\rho} + 1\right) K,$$

where  $w_{\max} = \max_j |w_j|$  and  $K$  is the universal constant in Lemma 3. Suppose the above statement is false, i.e., there exists  $i \in \{1, 2, \dots, n\}$  such that  $\hat{x}_i^t \neq x_i^*$  for  $t > \left(\frac{w_{\max}}{\rho} + 1\right) K$ . Under the assumption, we will reach a contradiction.

Now we construct a tree-structured GM  $T_i(t)$ , popularly known as the computational tree (Weiss and Freeman, 2001), as follows:

1. Add  $y_i \in \{0, 1\}$  as the root variable with variable factor function  $e^{-w_i y_i}$ .
2. For each leaf variable  $y_j$  and for each  $\alpha \in F_j$  and  $\psi_\alpha$  is not associated with  $y_j$  in the current tree-structured GM, add a factor function  $\psi_\alpha$  as a child of  $y_j$ .
3. For each leaf factor  $\psi_\alpha$  and for each variable  $y_k$  such that  $k \in \alpha$  and  $y_k$  is not associated with  $\psi_\alpha$  in the current tree-structured GM, add a variable  $y_k$  as a child of  $\psi_\alpha$  with variable factor function  $e^{-w_k y_k}$ .

4. Repeat Step 2, 3  $t$  times.

Suppose the initial messages of BP are set by 1, i.e.,  $m_{j \rightarrow \alpha}(\cdot)^0 = 1$ . Then, if  $x_i^* \neq \hat{x}_i^t$ , it is known (Weiss, 1997) that there exists a MAP configuration  $y^{MAP}$  on  $T_i(t)$  with  $y_i^{MAP} \neq x_i^*$  at the root variable. For other initial messages, one can guarantee the same property under changing weights of leaf variables of the tree-structured GM. Specifically, for a leaf variable  $k$  with  $|F_k = \{\alpha_1, \alpha_2\}| = 2$  and  $\alpha_1$  being its parent factor in  $T_i(t)$ , we reset its variable factor by  $e^{-w_k y_k}$ , where

$$w'_k = w_k - \log \frac{\max_{z_{\alpha_2}: z_k=1} \psi_{\alpha_2}(z_{\alpha_2}) \prod_{j \in \alpha_2 \setminus k} m_{j \rightarrow \alpha_2}^0(z_j)}{\max_{z_{\alpha_2}: z_k=0} \psi_{\alpha_2}(z_{\alpha_2}) \prod_{j \in \alpha_2 \setminus k} m_{j \rightarrow \alpha_2}^0(z_j)}. \quad (7)$$

This is the reason why our proof of Theorem 1 goes through for arbitrary initial messages. For notational convenience, we present the proof for the standard initial message of  $m_{j \rightarrow \alpha}^0(\cdot) = 1$ , where it can be naturally generalized to other initial messages using (7).

Now we construct a new valid assignment  $y^{NEW}$  on the computational tree  $T_i(t)$  as follows:

1. Initially, set  $y^{NEW} \leftarrow y^{MAP}$ .
2. Update the value of the root variable of  $T_i(t)$  by  $y_i^{NEW} \leftarrow x_i^*$ .
3. For each child factor  $\psi_\alpha$  of root  $i \in \alpha$ , choose  $\gamma \subset \alpha$  according to Condition C3 and update the associated variable by  $y_j^{NEW} \leftarrow x_j^* \forall j \in \gamma$ .
4. Repeat Step 2,3 recursively by substituting  $T_i(t)$  by the subtree of  $T_i(t)$  of root  $j \in \gamma$  until the process stops (i.e.,  $i = j$ ) or the leaf of  $T_i(t)$  is reached (i.e.,  $i$  does not have a child).

One can notice that the set of revised variables in Step 2 of the above procedure forms a path structure  $Q$  in the tree-structured GM. We first, consider the case that both ends of the path  $Q$  touch leaves of  $T_i(t)$ , where other cases can be argued in a similar manner. Define  $\zeta_j$  and  $\kappa_j$  be the number of copies of  $x_j$  in path  $Q$  with  $x_j^* = 1$  and  $x_j^* = 0$ , respectively, where  $\zeta = [\zeta_j], \kappa = [\kappa_j] \in \mathbb{Z}_+^n$ . Then, from our construction of  $y^{NEW}$ , one can observe that

$$y^{NEW} = y^{MAP} + \zeta - \kappa$$

$$w \cdot y^{MAP} - w \cdot y^{NEW} = w \cdot (\kappa - \zeta).$$

If we set  $z = x^* + \varepsilon(\kappa - \zeta)$  where  $0 < \varepsilon < \min\{1/2t, \eta\}$ , then one can check that  $z$  satisfies the conditions of Lemma 3 using Conditions C2, C3. Hence, from Lemma 3, there exists  $z^\ddagger \in \mathcal{P}$  such that

$$\|z^\ddagger - z\|_1 \leq \varepsilon K$$

$$\|z^\ddagger - x^*\|_1 \geq \varepsilon(\|\zeta\|_1 + \|\kappa\|_1 - K) \geq \varepsilon(t - K).$$

where  $z = x^* + \varepsilon(\kappa - \zeta)$ . Hence, it follows that

$$\begin{aligned} 0 < \rho &\leq \frac{w \cdot z^\dagger - w \cdot x^*}{\|z^\dagger - x^*\|_1} \\ &\leq \frac{w \cdot z + \varepsilon w_{\max} K - w \cdot x^*}{\varepsilon(t - K)} \\ &= \frac{\varepsilon w \cdot (\kappa - \zeta) + \varepsilon w_{\max} K}{\varepsilon(t - K)} \\ &= \frac{w \cdot (\kappa - \zeta) + w_{\max} K}{t - K} \end{aligned}$$

Furthermore, if  $t > \left(\frac{w_{\max}}{\rho} + 1\right) K$ , the above inequality implies that

$$\begin{aligned} w \cdot y^{MAP} - w \cdot y^{NEW} &= w \cdot (\kappa - \zeta) \\ &\geq \rho t - (w_{\max} + \rho) K > 0. \end{aligned}$$

This contradicts to the fact that  $y^{MAP}$  is a MAP configuration. This completes the proof of Theorem 1.

### 3.2 PROOF OF LEMMA 3

One can write  $\mathcal{P} = \{x : Ax \geq b\} \subset [0, 1]^n$  for some matrix  $A \in \mathbb{R}^{m \times n}$  and vector  $b \in \mathbb{R}^m$ , where without loss of generality, we can assume that  $\|A_i\|_2 = 1$  where  $\{A_i\}$  is the set of row vectors of  $A$ . We define

$$\mathcal{P}_\varepsilon = \{x : Ax \geq b - \varepsilon \mathbf{1}\},$$

where  $\mathbf{1}$  is the vector of ones. Then, one can check that  $z \in \mathcal{P}_\varepsilon$  for  $z, \varepsilon$  satisfying conditions of Lemma 3. Now we aim for finding a universal constant  $K$  satisfying

$$\text{dist}(\mathcal{P}, \mathcal{P}_\varepsilon) := \max_{x \in \mathcal{P}_\varepsilon} (\min_{y \in \mathcal{P}} \|x - y\|_1) \leq \varepsilon K,$$

which leads to the conclusion of Lemma 3.

To this end, for  $\xi \subset [1, 2, \dots, m]$  with  $|\xi| = n$ , we let  $A_\xi$  be the square sub-matrix of  $A$  by choosing  $\xi$ -th rows of  $A$  and  $b_\xi$  is the  $n$ -dimensional subvector of  $b$  corresponding  $\xi$ . Throughout the proof, we only consider  $\xi$  such that  $A_\xi$  is invertible. Using this notation, we first claim the following.

**Claim 4** *If  $A_\xi$  is invertible and  $v_\xi := A_\xi^{-1} b_\xi \in \mathcal{P}$ , then  $v_\xi$  is a vertex of polytope  $\mathcal{P}$ .*

**Proof.** Suppose  $v_\xi$  is not a vertex of  $\mathcal{P}$ , i.e. there exist  $x, y \in \mathcal{P}$  such that  $x \neq y$  and  $v_\xi = \lambda x + (1 - \lambda)y$  for some  $\lambda \in (0, 1/2]$ . Under the assumption, we will reach a contradiction. Since  $\mathcal{P}$  is a convex set,

$$\frac{3\lambda}{2}x + \left(1 - \frac{3\lambda}{2}\right)y \in \mathcal{P}. \quad (8)$$

However, as  $A_\xi$  is invertible,

$$A_\xi \left( \frac{3\lambda}{2}x + \left(1 - \frac{3\lambda}{2}\right)y \right) \neq b_\xi. \quad (9)$$

From (8) and (9), there exists a row vector  $A_i$  of  $A_\xi$  and the corresponding element  $b_i$  of  $b_\xi$  such that

$$A_i \cdot \left( \frac{3\lambda}{2}x + \left(1 - \frac{3\lambda}{2}\right)y \right) > b_i.$$

Using the above inequality and  $A_i \cdot (\lambda x + (1 - \lambda)y) = b_i$ , one can conclude that

$$A_i \cdot \left( \frac{\lambda}{2}x + \left(1 - \frac{\lambda}{2}\right)y \right) < b_i,$$

which contradict to  $\frac{\lambda}{2}x + \left(1 - \frac{\lambda}{2}\right)y \in \mathcal{P}$ . This completes the proof of Claim 4.  $\square$

We also note that if  $v$  is a vertex of polytope  $\mathcal{P}$ , there exists  $\xi$  such that  $A_\xi$  is invertible and  $v = A_\xi^{-1} b_\xi$ . We define the following notation:

$$\mathcal{I} = \{\xi : A_\xi^{-1} b_\xi \in \mathcal{P}\} \quad \mathcal{I}_\varepsilon = \{\xi : A_\xi^{-1} (b_\xi - \varepsilon \mathbf{1}) \in \mathcal{P}_\varepsilon\},$$

where Claim 4 implies that  $\{v_\xi := A_\xi^{-1} b_\xi : \xi \in \mathcal{I}\}$  and  $\{u_{\xi, \varepsilon} := A_\xi^{-1} (b_\xi - \varepsilon \mathbf{1}) : \xi \in \mathcal{I}_\varepsilon\}$  are sets of vertices of  $\mathcal{P}$  and  $\mathcal{P}_\varepsilon$ , respectively. Using the notation, we show the following claim.

**Claim 5** *There exists  $\eta > 0$  such that  $\mathcal{I}_\varepsilon \subset \mathcal{I}$  for all  $\varepsilon \in (0, \eta)$ .*

**Proof.** Suppose  $\eta > 0$  satisfying the conclusion of Claim 5 does not exist. Then, there exists a strictly decreasing sequence  $\{\varepsilon_k > 0 : k = 1, 2, \dots\}$  converges to 0 such that  $\mathcal{I}_{\varepsilon_k} - \mathcal{I} \neq \emptyset$ . Since  $|\{\xi : \xi \subset [1, 2, \dots, m]\}| < \infty$ , there exists  $\xi'$  such that

$$|\mathcal{K} := \{k : \xi' \in \mathcal{I}_{\varepsilon_k} - \mathcal{I}\}| = \infty. \quad (10)$$

For any  $k \in \mathcal{K}$ , observe that the sequence  $\{u_{\xi', \varepsilon_\ell} : \ell \geq k, \ell \in \mathcal{K}\}$  converges to  $v_{\xi'}$ . Furthermore, all points in the sequence are in  $\mathcal{P}_{\varepsilon_k}$  since  $\mathcal{P}_{\varepsilon_\ell} \subset \mathcal{P}_{\varepsilon_k}$  for any  $\ell \geq k$ . Therefore, one can conclude that  $v_{\xi'} \in \mathcal{P}_{\varepsilon_k}$  for all  $k \in \mathcal{K}$ , where we additionally use the fact that  $\mathcal{P}_{\varepsilon_k}$  is a closed set. Because  $\mathcal{P} = \bigcap_{k \in \mathcal{K}} \mathcal{P}_{\varepsilon_k}$ , it must be that  $v_{\xi'} \in \mathcal{P}$ , i.e.,  $v_{\xi'}$  must be a vertex of  $\mathcal{P}$  from Claim 4. This contradicts to the fact  $\xi' \notin \mathcal{I}$ . This completes the proof of Claim 5.  $\square$

From the above claim, we observe that any  $x \in \mathcal{P}_\varepsilon$  can be expressed as a convex combination of  $\{u_{\xi, \varepsilon} : \xi \in \mathcal{I}\}$ , i.e.,  $x = \sum_{\xi \in \mathcal{I}} \lambda_\xi u_{\xi, \varepsilon}$  with  $\sum_{\xi \in \mathcal{I}} \lambda_\xi = 1$  and  $\lambda_\xi \geq 0$ . For all  $\varepsilon \in (0, \eta)$  for  $\eta > 0$  in Claim 5, one can conclude that

$$\begin{aligned} \text{dist}(\mathcal{P}, \mathcal{P}_\varepsilon) &\leq \max_{x \in \mathcal{P}_\varepsilon} \left\| \sum_{\xi \in \mathcal{I}} \lambda_\xi u_{\xi, \varepsilon} - \sum_{\xi \in \mathcal{I}} \lambda_\xi v_\xi \right\|_1 \\ &= \max_{x \in \mathcal{P}_\varepsilon} \varepsilon \left\| \sum_{\xi \in \mathcal{I}} \lambda_\xi A_\xi^{-1} \mathbf{1} \right\|_1 \\ &\leq \varepsilon \max_{\xi} \|A_\xi^{-1} \mathbf{1}\|_1, \end{aligned}$$

where we choose  $K = \max_{\xi} \|A_\xi^{-1} \mathbf{1}\|_1$ . This completes the proof of Lemma 3.

## 4 APPLICATIONS OF THEOREM 1 TO COMBINATORIAL OPTIMIZATION

In this section, we introduce concrete instances of LPs satisfying the conditions of Theorem 1 so that BP correctly converges to its optimal solution. Specifically, we consider LP formulations associated to several combinatorial optimization problems including shortest path, maximum weight perfect matching, traveling salesman, maximum weight disjoint vertex cycle packing and vertex cover. We note that the shortest path result, Corollary 6, is known (Ruozzi and Tatikonda, 2008), where we rediscover it as a corollary of Theorem 1. Our other results, Corollaries 7-11, are new and what we first establish in this paper.

### 4.1 SHORTEST PATH

Given directed graph  $G = (V, E)$  and non-negative edge weights  $w = [w_e : e \in E] \in \mathbb{R}_+^{|E|}$ , the shortest path problem is to find the shortest path from the source  $s$  to the destination  $t$ : it minimizes the sum of edge weights along the path. One can naturally design the following LP for this problem:

$$\begin{aligned} & \text{minimize} && w \cdot x \\ & \text{subject to} && \sum_{e \in \delta^o(v)} x_e - \sum_{e \in \delta^i(v)} x_e \\ & && = \begin{cases} 1 & \text{if } v = s \\ -1 & \text{if } v = t \\ 0 & \text{otherwise} \end{cases} \quad \forall v \in V \\ & && x = [x_e] \in [0, 1]^{|E|}. \end{aligned} \quad (11)$$

where  $\delta^i(v), \delta^o(v)$  are the set of incoming, outgoing edges of  $v$ . It is known that the above LP always has an integral solution, i.e., the shortest path from  $s$  to  $t$ . We consider the following GM for LP (11):

$$\Pr[X = x] \propto \prod_{e \in E} e^{-w_e x_e} \prod_{v \in V} \psi_v(x_{\delta(v)}), \quad (12)$$

where the factor function  $\psi_v$  is defined as

$$\psi_v(x_{\delta(v)}) = \begin{cases} 1 & \text{if } \sum_{e \in \delta^o(v)} x_e - \sum_{e \in \delta^i(v)} x_e \\ & = \begin{cases} 1 & \text{if } v = s \\ -1 & \text{if } v = t \\ 0 & \text{otherwise} \end{cases} \\ 0 & \text{otherwise} \end{cases}.$$

For the above GM (12), one can easily check Conditions C2, C3 of Theorem 1 hold and derive the following corollary whose formal proof is presented in the supplementary material due to the space constraint.

**Corollary 6** *If the shortest path from  $s$  to  $t$ , i.e., the solution of the shortest path LP (11), is unique, then the max-product BP on GM (12) converges to it.*

The uniqueness condition in the above corollary is easy to guarantee by adding small random noises to edge weights.

### 4.2 MAXIMUM WEIGHT PERFECT MATCHING

Given undirected graph  $G = (V, E)$  and non-negative edge weights  $w = [w_e : e \in E] \in \mathbb{R}_+^{|E|}$  on edges, the maximum weight perfect matching problem is to find a set of edges such that each vertex is connected to exactly one edge in the set and the sum of edge weights in the set is maximized. One can naturally design the following LP for this problem:

$$\begin{aligned} & \text{maximize} && w \cdot x \\ & \text{subject to} && \sum_{e \in \delta(v)} x_e = 1, \quad \forall v \in V \\ & && x = [x_e] \in [0, 1]^{|E|}. \end{aligned} \quad (13)$$

where  $\delta(v)$  is the set of edges connected to a vertex  $v$ . If the above LP has an integral solution, it corresponds to the solution of the maximum weight perfect matching problem.

It is known that the maximum weight matching LP (13) always has a half-integral solution  $x^* \in \{0, \frac{1}{2}, 1\}^{|E|}$ . We will design BP for obtaining the half-integral solution. To this end, duplicate each edge  $e$  to  $e_1, e_2$  and define a new graph  $G' = (V, E')$  where  $E' = \{e_1, e_2 : e \in E\}$ . Then, we suggest the following equivalent LP that always have an integral solution:

$$\begin{aligned} & \text{maximize} && w' \cdot x \\ & \text{subject to} && \sum_{e_i \in \delta(v)} x_{e_i} = 2 \quad \forall v \in V \\ & && x = [x_{e_i}] \in [0, 1]^{|E'|}. \end{aligned} \quad (14)$$

where  $w'_{e_1} = w'_{e_2} = w_e$ . One can easily observe that solving LP (14) is equivalent to solving LP (13) due to our construction of  $G'$  and  $w'$ . Now, construct the following GM for LP (14):

$$\Pr[X = x] \propto \prod_{e_i \in E'} e^{w'_{e_i} x_{e_i}} \prod_{v \in V} \psi_v(x_{\delta(v)}), \quad (15)$$

where the factor function  $\psi_v$  is defined as

$$\psi_v(x_{\delta(v)}) = \begin{cases} 1 & \text{if } \sum_{e_i \in \delta(v)} x_{e_i} = 2 \\ 0 & \text{otherwise} \end{cases}.$$

For the above GM (15), we derive the following corollary of Theorem 1 whose formal proof is presented in the supplementary material due to the space constraint.

**Corollary 7** *If the solution of the maximum weight perfect matching LP (14) is unique, then the max-product BP on GM (15) converges to it.*

Again, the uniqueness condition in the above corollary is easy to guarantee by adding small random noises to edge weights  $[w'_{e_i}]$ . We note that it is known (Bayati et al., 2011) that BP converges to the unique and integral solution of LP (13), while Corollary 7 implies that BP can solve it without the integrality condition. We note that one can easily obtain a similar result for the maximum weight (non-perfect) matching problem, where we omit the details in this paper.

### 4.3 MAXIMUM WEIGHT PERFECT MATCHING WITH ODD CYCLES

In previous section we prove that BP converges to the optimal (possibly, fractional) solution of LP (14), equivalently LP (13). One can add odd cycle (also called Blossom) constraints and make those LPs tight i.e. solves the maximum weight perfect matching problem:

$$\begin{aligned} & \text{maximize} && w \cdot x \\ & \text{subject to} && \sum_{e \in \delta(v)} x_e = 1, \quad \forall v \in V \\ & && \sum_{e \in C} x_e \leq \frac{|C| - 1}{2}, \quad \forall C \in \mathcal{C}, \\ & && x = [x_e] \in [0, 1]^{|E|}. \end{aligned} \quad (16)$$

where  $\mathcal{C}$  is a set of odd cycles in  $G$ . The authors (Shin et al., 2013) study BP for solving LP (16) by replacing  $\sum_{e \in \delta(v)} x_e = 1$  by  $\sum_{e \in \delta(v)} x_e \leq 1$ , i.e., for the maximum weight (non-perfect) matching problem. Using Theorem 1, one can extend the result to the maximum weight perfect matching problem, i.e., solving LP (16). To this end, we follow the approach (Shin et al., 2013) and construct the following graph  $G' = (V', E')$  and weight  $w' = [w'_e : e \in E'] \in \mathbb{R}^{|E'|}$  given set  $\mathcal{C}$  of disjoint odd cycles:

$$\begin{aligned} V' &= V \cup \{v_C : C \in \mathcal{C}\} \\ E' &= \{(u, v_C) : u \in C, C \in \mathcal{C}\} \cup E \setminus \{e \in C : C \in \mathcal{C}\} \end{aligned}$$

$$w'_e = \begin{cases} \frac{1}{2} \sum_{e' \in E(C)} (-1)^{d_C(u, e')} w_{e'} & \text{if } e = (u, v_C) \\ & \text{for some } C \in \mathcal{C}, \\ w_e & \text{otherwise} \end{cases}$$

where  $d_C(u, e')$  is the graph distance between  $u, e'$  in cycle  $C$ . Then, LP (16) is equivalent to the following LP:

$$\begin{aligned} & \text{maximize} && w' \cdot y \\ & \text{subject to} && \sum_{e \in \delta(v)} y_e = 1, \quad \forall v \in V \\ & && \sum_{u \in V(C)} (-1)^{d_C(u, e)} y_{(v_C, u)} \in [0, 2], \quad \forall e \in E(C) \\ & && \sum_{e \in \delta(v_C)} y_e \leq |C| - 1, \quad \forall C \in \mathcal{C} \\ & && y = [y_e] \in [0, 1]^{|E'|}. \end{aligned} \quad (17)$$

Now, we construct the following GM from the above LP:

$$\Pr[Y = y] \propto \prod_{e \in E} e^{w_e y_e} \prod_{v \in V} \psi_v(y_{\delta(v)}) \prod_{C \in \mathcal{C}} \psi_C(y_{\delta(v_C)}), \quad (18)$$

where the factor function  $\psi_v, \psi_C$  is defined as

$$\begin{aligned} \psi_v(y_{\delta(v)}) &= \begin{cases} 1 & \text{if } \sum_{e \in \delta(v)} y_e = 1 \\ 0 & \text{otherwise} \end{cases}, \\ \psi_C(y_{\delta(v_C)}) &= \begin{cases} 1 & \text{if } \sum_{u \in V(C)} (-1)^{d_C(u, e)} y_{(v_C, u)} \in \{0, 2\} \\ & \sum_{e \in \delta(v_C)} y_e \leq |C| - 1 \\ 0 & \text{otherwise} \end{cases}. \end{aligned}$$

For the above GM (18), we derive the following corollary of Theorem 1 whose formal proof is presented in the supplementary material due to the space constraint.

**Corollary 8** *If the solution of the maximum weight perfect matching with odd cycles LP (17) is unique and integral, then the max-product BP on GM (18) converges to it.*

We again emphasize that a similar result for the maximum weight (non-perfect) matching problem was established in (Shin et al., 2013). However, the proof technique in the paper does not extend to the perfect matching problem. This is in essence because presumably the perfect matching problem is harder than the non-perfect matching one. Under the proposed generic criteria of Theorem 1, we overcome the technical difficulty.

### 4.4 VERTEX COVER

Given undirected graph  $G = (V, E)$  and non-negative integer vertex weights  $b = [b_v : v \in V] \in \mathbb{Z}_+^{|V|}$ , the vertex cover problem is to find a set of vertices minimizes the sum of vertex weights in the set such that each edge is connected to at least one vertex in it. This problem is one of Karp's 21 NP-complete problems (Karp, 1972). The associated LP formulation to the vertex cover problem is as follows:

$$\begin{aligned} & \text{minimize} && b \cdot y \\ & \text{subject to} && y_u + y_v \geq 1, (u, v) \in E \\ & && y = [y_v] \in [0, 1]^{|V|}. \end{aligned} \quad (19)$$

However, if we design a GM from the above LP, it does not satisfy conditions in Theorem 1. Instead, we will show that BP can solve the following dual LP:

$$\begin{aligned} & \text{maximize} && \sum_{e \in E} x_e \\ & \text{subject to} && \sum_{e \in \delta(v)} x_e \leq b_v, \quad \forall v \in V \\ & && x = [x_e] \in \mathbb{R}_+^{|E|}. \end{aligned} \quad (20)$$

Note that the above LP always has a half-integral solution. As we did in Section 4.2, one can duplicate edges, i.e.,

$E' = \{e_1, \dots, e_{2b_{\max}} : e \in E\}$  with  $b_{\max} = \max_v b_v$ , and design the following equivalent LP having an integral solution:

$$\begin{aligned} & \text{maximize} && w' \cdot x \\ & \text{subject to} && \sum_{e_i \in \delta(v)} x_{e_i} \leq 2b_v, \quad \forall v \in V, \\ & && x = [x_{e_i}] \in [0, 1]^{|E'|} \end{aligned} \quad (21)$$

where  $w'_{e_i} = w_e$  for  $e \in E$  and its copy  $e_i \in E'$ . From the above LP, we can construct the following GM:

$$\Pr[X = x] \propto \prod_{e_i \in E'} e^{w'_{e_i} x_{e_i}} \prod_{v \in V} \psi_v(x_{\delta(v)}), \quad (22)$$

where the factor function  $\psi_v$  is defined as

$$\psi_v(x_{\delta(v)}) = \begin{cases} 1 & \text{if } \sum_{e_i \in \delta(v)} x_{e_i} \leq 2b_v \\ 0 & \text{otherwise} \end{cases}.$$

For the above GM (22), we derive the following corollary of Theorem 1 whose formal proof is presented in the supplementary material due to the space constraint.

**Corollary 9** *If the solution of the vertex cover dual LP (21) is unique, then the max-product BP on GM (22) converges it.*

Again, the uniqueness condition in the above corollary is easy to guarantee by adding small random noises to edge weights  $[w'_{e_i}]$ . We further remark that if the solution of the primal LP (19) is integral, then it can be easily found from the solution of the dual LP (21) using the strictly complementary slackness condition (Bertsimas and Tsitsiklis, 1997).

#### 4.5 TRAVELING SALESMAN

Given directed graph  $G = (V, E)$  and non-negative edge weights  $w = [w_e : e \in E] \in \mathbb{R}_+^{|E|}$ , the traveling salesman problem (TSP) is to find the minimum weight Hamiltonian cycle in  $G$ . The natural LP formulation to TSP is following:

$$\begin{aligned} & \text{minimize} && w \cdot x \\ & \text{subject to} && \sum_{e \in \delta(v)} x_e = 2 \\ & && x = [x_e] \in [0, 1]^{|E|}. \end{aligned} \quad (23)$$

From the above LP, one can construct the following GM:

$$\Pr[X = x] \propto \prod_{e \in E} e^{-w_e x_e} \prod_{v \in V} \psi_v(x_{\delta(v)}), \quad (24)$$

where the factor function  $\psi_v$  is defined as

$$\psi_v(x_{\delta(v)}) = \begin{cases} 1 & \text{if } \sum_{e \in \delta(v)} x_e = 2 \\ 0 & \text{otherwise} \end{cases}.$$

For the above GM (24), we derive the following corollary of Theorem 1 whose formal proof is presented in the supplementary material due to the space constraint.

**Corollary 10** *If the solution of the traveling salesman LP (23) is unique and integral, then the max-product BP on GM (24) converges it.*

Again, the uniqueness condition in the above corollary is easy to guarantee by adding small random noises to edge weights. In Section 5, we show the empirical performance of the max-product BP on GM (24) for solving TSP without relying on the integrality condition in Corollary 10.

#### 4.6 MAXIMUM WEIGHT CYCLE PACKING

Given undirected graph  $G = (V, E)$  and non-negative edge weights  $w = [w_e : e \in E] \in \mathbb{R}_+^{|E|}$ , the maximum weight vertex disjoint cycle packing problem is to find the maximum weight set of cycles with no common vertex. It is easy to observe that it is equivalent to find a subgraph maximizing the sum of edge weights on it such that each vertex of the subgraph has degree 2 or 0. The natural LP formulation to this problem is following:

$$\begin{aligned} & \text{maximize} && w \cdot x \\ & \text{subject to} && \sum_{e \in \delta(v)} x_e = 2y_v \\ & && x = [x_e] \in [0, 1]^{|E|}, y = [y_v] \in [0, 1]^{|V|}. \end{aligned} \quad (25)$$

From the above LP, one can construct the following GM:

$$\Pr[X = x, Y = y] \propto \prod_{e \in E} e^{w_e x_e} \prod_{v \in V} \psi_v(x_{\delta(v)}, y_v), \quad (26)$$

where the factor function  $\psi_v$  is defined as

$$\psi_v(x_{\delta(v)}, y_v) = \begin{cases} 1 & \text{if } \sum_{e \in \delta(v)} x_e = 2y_v \\ 0 & \text{otherwise} \end{cases}.$$

For the above GM (26), we derive the following corollary of Theorem 1 whose formal proof is presented in the supplementary material due to the space constraint.

**Corollary 11** *If the solution of maximum weight vertex disjoint cycle packing LP (25) is unique and integral, then the max-product BP on GM (26) converges it.*

Again, the uniqueness condition in the above corollary is easy to guarantee by adding small random noises to edge weights.

### 5 EXPERIMENTAL RESULTS FOR TRAVELING SALESMAN PROBLEM

In this section, we report empirical performances of BP on GM (24) for solving the traveling salesman problem (TSP)

Table 1: Experimental results for small size complete graph and each number is the average among 100 samples. For example, Greedy+BP means that the Greedy algorithm using edge weights as BP beliefs as we describe in Section 5. The left value is the approximation ratio, i.e., the average weight ratio between the heuristic solution and the exact solution. The right value is the average weight of the heuristic solutions. The last row is a ratio of tight TSP LP (23).

| Size            | 5           | 10          | 15          | 20          | 25          |
|-----------------|-------------|-------------|-------------|-------------|-------------|
| Greedy          | 1.07 / 1.84 | 1.20 / 2.25 | 1.33 / 2.58 | 1.51 / 2.85 | 1.51 / 3.04 |
| Greedy+BP       | 1.00 / 1.75 | 1.05 / 1.98 | 1.13 / 2.23 | 1.19 / 2.27 | 1.21 / 2.43 |
| Christofides    | 1.38 / 1.85 | 1.38 / 2.56 | 1.67 / 3.20 | 1.99 / 3.75 | 2.16 / 4.32 |
| Christofides+BP | 1.00 / 1.75 | 1.09 / 2.07 | 1.23 / 2.43 | 1.30 / 2.50 | 1.45 / 2.90 |
| Insertion       | 1.03 / 1.79 | 1.29 / 2.38 | 1.53 / 2.95 | 1.72 / 3.26 | 1.89 / 3.77 |
| Insertion+BP    | 1.00 / 1.75 | 1.29 / 2.39 | 1.52 / 2.97 | 1.79 / 3.38 | 1.94 / 3.89 |
| N-Neighbor      | 1.07 / 1.84 | 1.27 / 2.39 | 1.42 / 2.74 | 1.55 / 2.96 | 1.64 / 3.30 |
| N-Neighbor+BP   | 1.00 / 1.75 | 1.05 / 1.98 | 1.13 / 2.23 | 1.15 / 2.21 | 1.20 / 2.40 |
| 2-Opt           | 1.00 / 1.75 | 1.08 / 2.04 | 1.12 / 2.21 | 1.24 / 2.36 | 1.28 / 2.57 |
| 2-Opt+BP        | 1.00 / 1.75 | 1.04 / 1.96 | 1.07 / 2.11 | 1.11 / 2.13 | 1.16 / 2.34 |
| Tight LPs       | 100%        | 93%         | 88%         | 87%         | 84%         |

Table 2: Experimental results for sparse Erdos-Renyi graph with fixed average vertex degrees and each number is the average among 1000 samples. The left value is the ratio that a heuristic finds the Hamiltonian cycle without penalty edges. The right value is the average weight of the heuristic solutions.

| Size            | 100           |                |                | 200            |                |                |
|-----------------|---------------|----------------|----------------|----------------|----------------|----------------|
|                 | 10            | 25             | 50             | 10             | 25             | 50             |
| Greedy          | 0% / 7729.43  | 0.3% / 2841.98 | 13% / 1259.08  | 0% / 15619.9   | 0% / 5828.88   | 0.3% / 2766.07 |
| Greedy+BP       | 14% / 1612.82 | 21% / 1110.27  | 44% / 622.488  | 6.4% / 2314.95 | 10% / 1687.29  | 16% / 1198.48  |
| Christofides    | 0% / 19527.3  | 0% / 16114.3   | 0% / 10763.7   | 0% / 41382.5   | 0% / 37297.0   | 0% / 32023.1   |
| Christofides+BP | 14% / 2415.73 | 20% / 1663.47  | 34% / 965.775  | 6.1% / 3586.77 | 9.2% / 2876.35 | 12% / 2183.80  |
| Insertion       | 0% / 12739.2  | 84% / 198.099  | 100% / 14.2655 | 0% / 34801.6   | 0.9% / 3780.71 | 99% / 44.1293  |
| Insertion+BP    | 0% / 13029.0  | 76% / 283.766  | 100% / 14.6964 | 0% / 34146.7   | 0.3% / 4349.11 | 99% / 41.2176  |
| N-Neighbor      | 0% / 9312.77  | 0% / 3385.14   | 7.6% / 1531.83 | 0% / 19090.7   | 0% / 7383.23   | 0.3% / 3484.82 |
| N-Neighbor+BP   | 16% / 1206.95 | 26% / 824.232  | 50% / 509.349  | 6.9% / 1782.17 | 12% / 1170.38  | 24% / 888.421  |
| 2-Opt           | 34% / 1078.03 | 100% / 14.6873 | 100% / 7.36289 | 2% / 3522.78   | 100% / 35.8421 | 100% / 18.6147 |
| 2-Opt+BP        | 76% / 293.450 | 100% / 13.5773 | 100% / 6.53995 | 33% / 1088.79  | 100% / 34.7768 | 100% / 17.4883 |
| Tight LPs       | 62%           | 62.3%          | 63%            | 52.2%          | 55%            | 52.2%          |

that is presumably the most famous one in combinatorial optimization. In particular, we design the following BP-guided heuristic for solving TSP:

1. Run BP for a fixed number of iterations, say 100, and calculate the BP marginal beliefs (3).
2. Run the known TSP heuristic using edge weights as  $\log \frac{b_{[0]}}{b_{[1]}}$  using BP marginal beliefs instead of the original weights.

For TSP heuristic in Step 2, we use Greedy, Christofides, Insertion, N-Neighbor and 2-Opt provided by the LEMON graph library (Dezso et al., 2011). We first perform the experiments on the complete graphs of size 5, 10, 15, 20, 25 and random edge weight in  $(0, 1)$  to measure approximation qualities of heuristics, where it is reported in Table 1. Second, we consider the sparse Erdos-Renyi random graph of size 100, 200 and random edge weight in  $(0, 1)$ . Then, we make it a complete graph by adding non-existing edges with penalty edge weight 1000.<sup>2</sup> For these random in-

<sup>2</sup>This is to ensure that a Hamiltonian cycle always exists.

stances, we report performance of various heuristics in Table 2. Our experiments show that BP boosts performances of known TSP heuristics in overall, where BP is very easy to code and does not hurt the simplicity of heuristics.

## 6 CONCLUSION

The BP algorithm has been the most popular algorithm for solving inference problems arising graphical models, where its distributed implementation, associated ease of programming and strong parallelization potential are the main reasons for its growing popularity. In this paper, we aim for designing BP algorithms solving LPs, and provide sufficient conditions for its correctness and convergence. We believe that our results provide new interesting directions on designing efficient distributed (and parallel) solvers for large-scale LPs.

### Acknowledgements.

We would like to acknowledge the support of the AOARD project, FA2386-14-1-4058.



## References

- Mohsen Bayati, Devavrat Shah, and Mayank Sharma. Maximum weight matching via max-product belief propagation. In *Information Theory, 2005. ISIT 2005. Proceedings. International Symposium on*, pages 1763–1767. IEEE, 2005.
- Mohsen Bayati, Christian Borgs, Jennifer Chayes, and Riccardo Zecchina. Belief propagation for weighted b-matchings on arbitrary graphs and its relation to linear programs with integer solutions. *SIAM Journal on Discrete Mathematics*, 25(2):989–1011, 2011.
- Dimitris Bertsimas and John N Tsitsiklis. *Introduction to linear optimization*, volume 6. Athena Scientific Belmont, MA, 1997.
- Venkat Chandrasekaran, Nathan Srebro, and Prahladh Harsha. Complexity of inference in graphical models. In *UAI 2008, Proceedings of the 24th Conference in Uncertainty in Artificial Intelligence*, pages 70–78. AUAI Press, 2008.
- George B Dantzig. *Linear programming and extensions*. Princeton university press, 1998.
- Balázs Dezső, Alpár Jüttner, and Péter Kovács. Lemon—an open source c++ graph template library. *Electronic Notes in Theoretical Computer Science*, 264(5):23–45, 2011.
- Joseph Gonzalez, Yucheng Low, and Carlos Guestrin. Parallel splash belief propagation. Technical report, DTIC Document, 2010.
- Bert C Huang and Tony Jebara. Loopy belief propagation for bipartite maximum weight b-matching. In *International Conference on Artificial Intelligence and Statistics*, pages 195–202, 2007.
- Richard M Karp. *Reducibility among combinatorial problems*. Springer, 1972.
- Leonid G Khachiyan. Polynomial algorithms in linear programming. *USSR Computational Mathematics and Mathematical Physics*, 20(1):53–72, 1980.
- Yucheng Low, Joseph Gonzalez, Aapo Kyrola, Danny Bickson, Carlos Guestrin, and Joseph M. Hellerstein. Graphlab: A new framework for parallel machine learning. In *UAI 2010, Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, pages 340–349. AUAI Press, 2010.
- Nam Ma, Yinglong Xia, and Viktor K Prasanna. Task parallel implementation of belief propagation in factor graphs. In *Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2012 IEEE 26th International*, pages 1944–1953. IEEE, 2012.
- Marc Mezard and Andrea Montanari. *Information, physics, and computation*. Oxford University Press, 2009.
- Tom Richardson and Ruediger Urbanke. *Modern coding theory*. Cambridge University Press, 2008.
- Nicholas Ruozi and Sekhar Tatikonda. st paths using the min-sum algorithm. In *Communication, Control, and Computing, 2008 46th Annual Allerton Conference on*, pages 918–921. IEEE, 2008.
- Justin Salez and Devavrat Shah. Belief propagation: an asymptotically optimal algorithm for the random assignment problem. *Mathematics of Operations Research*, 34(2):468–480, 2009.
- Sujay Sanghavi, Dmitry Malioutov, and Alan Willsky. Belief propagation and lp relaxation for weighted matching in general graphs. *Information Theory, IEEE Transactions on*, 57(4):2203–2212, 2011.
- Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer, 2003.
- Jinwoo Shin, Andrew E Gelfand, and Misha Chertkov. A graphical transformation for belief propagation: Maximum weight matchings and odd-sized cycles. In *Advances in Neural Information Processing Systems*, pages 2022–2030, 2013.
- George B Dantzig Mukund N Thapa. Linear programming. 2003.
- Meritxell Vinyals, Alessandro Farinelli, Juan A Rodríguez-aguilar, et al. Worst-case bounds on the quality of max-product fixed-points. In *Advances in Neural Information Processing Systems*, pages 2325–2333, 2010.
- Martin J Wainwright and Michael I Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1-2): 1–305, 2008.
- Yair Weiss. Belief propagation and revision in networks with loops. 1997.
- Yair Weiss and William T Freeman. On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs. *Information Theory, IEEE Transactions on*, 47(2):736–744, 2001.
- Jonathan S Yedidia, William T Freeman, and Yair Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *Information Theory, IEEE Transactions on*, 51(7):2282–2312, 2005.

---

# Fast Algorithms for Learning with Long $N$ -grams via Suffix Tree Based Matrix Multiplication

---

**Hristo S. Paskov**  
Computer Science Dept.  
Stanford University  
hpaskov@stanford.edu

**John C. Mitchell**  
Computer Science Dept.  
Stanford University  
john.mitchell@stanford.edu

**Trevor J. Hastie**  
Statistics Dept.  
Stanford University  
hastie@stanford.edu

## Abstract

This paper addresses the computational issues of learning with long, and possibly all,  $N$ -grams in a document corpus. Our main result uses suffix trees to show that  $N$ -gram matrices require memory and time that is at worst *linear* (in the length of the underlying corpus) to store and to multiply a vector. Our algorithm can speed up any  $N$ -gram based machine learning algorithm which uses gradient descent or an optimization procedure that makes progress through multiplication. We also provide a linear running time and memory framework that screens  $N$ -gram features according to a multitude of statistical criteria and produces the data structure necessary for fast multiplication. Experiments on natural language and DNA sequence datasets demonstrate the computational savings of our framework; our multiplication algorithm is four orders of magnitude more efficient than naïve multiplication on the DNA data. We also show that prediction accuracy on large-scale sentiment analysis problems benefits from long  $N$ -grams.

## 1 Introduction

$N$ -gram models are indispensable in natural language processing (NLP), information retrieval, and, increasingly, computational biology. They have been applied successfully in sentiment analysis (Paskov, 2013), text categorization (Cavnar, 1994), author identification (Houvardas, 2006), DNA function prediction (Kähärä, 2013), and numerous other tasks. The allure of  $N$ -gram models comes from their simplicity, interpretability, and efficacy: a document corpus is represented by its  $N$ -gram matrix, where each row/column corresponds to a distinct document/ $N$ -gram respectively, and each entry counts the number of occurrences of that  $N$ -gram in the document. The  $N$ -gram matrix provides a feature representation for statistical mod-

elling and the coefficients of each  $N$ -gram can often be interpreted as a score indicating their relevance to the task.

At the simplest extreme, unigrams provide a summary of the word distribution in each document and serve as an effective baseline representation for a variety of NLP tasks. Higher order  $N$ -grams provide more nuance by capturing short-term positional information and can achieve state of the art results on a variety of tasks (Wang, 2012) (Paskov, 2013). A canonical example of the value of longer  $N$ -grams is given by the phrase "I don't like horror movies, but this was excellent," which fails to convey its positive sentiment when its words are scrambled. Unfortunately, this additional information comes at a cost: a document of  $n$  words may contain up to  $\Theta(Kn)$  *distinct*  $N$ -grams of length  $K$ <sup>1</sup>. This growth makes the memory and computational burden of training  $N$ -gram models beyond bigrams impractical for large natural language corpora. Statistically, these larger feature representations suffer from the curse of dimensionality (Hastie, 2001) and may lead the model to overfit, so careful regularization is necessary.

This paper ameliorates the computational burden of learning with long  $N$ -grams. We demonstrate how the structure of suffix trees can be used to store and multiply<sup>2</sup> any  $N$ -gram matrix in time and space that is at most *linear* in the length of the underlying corpus. As most learning algorithms rely on matrix-vector multiplication to learn and predict, our results equate the computational cost of learning with  $N$ -gram matrices to scanning through the original corpus. Our method can speed up *any* learning algorithm that exhibits such structure by simply replacing its multiplication routine with ours. Fast multiplication is possible by means of a specialized data structure that efficiently represents the algebraic structure of the  $N$ -gram matrix. In view of the statistical issues associated with long  $N$ -grams, we provide a linear running time and memory framework that not only computes this data structure, but also filters  $N$ -grams by various criteria and outputs necessary column

---

<sup>1</sup>We use  $N$ -grams of length  $K$  to mean  $N$ -grams of length *at most*  $K$  for brevity.

<sup>2</sup>Multiplication always refers to matrix-vector multiplication.

normalizations. The emphasis of this framework is minimality: by only storing the topological structure of the suffix tree we achieve memory requirements that are comparable to storing the original document corpus. As such, our framework can be used to permanently store the corpus in a format that is optimized for machine learning.

Our paper is organized as follows: section 2 derives the fast multiplication algorithm by showing that after redundant columns in the  $N$ -gram matrix are removed, the algebraic structure of the resulting submatrix is encoded by the suffix tree of the underlying corpus. We then investigate how this matrix can be used as a black-box in various common learning scenarios in section 3. Section 4 presents our preprocessing framework. Timing and memory benchmarks that demonstrate the efficacy of the multiplication algorithm are presented in section 5. We also find that high-order  $N$ -grams can improve prediction accuracy in large-scale sentiment analysis tasks.

### 1.1 Related Work

Suffix trees and arrays are used by (Vish., 2004), (Teo, 2006), and (Rieck, 2008) for kernels that efficiently compute pair-wise document similarities based on  $N$ -grams. Computing the similarity of all document pairs limits kernels to moderately sized datasets and the lack of explicit features prevents the use of sparsity inducing regularizers such as in the Lasso (Tibs., 1996). Next, (Zhang, 2006) use suffix trees to identify useful  $N$ -grams in a text corpus and to show that the all  $N$ -gram matrix may be pruned since it contains redundant columns. We show in section 2 that the resulting  $N$ -gram matrix may still have too many entries to be practical for large corpora and observe this experimentally. Suffix trees are also used by (Wood, 2011) to efficiently represent and perform inference with a hierarchical process for text. Finally, while (Abou., 2004) and (Kasai, 2001) provide space efficient frameworks for working with suffix arrays, our framework is specialized to statistical processing and achieves greater memory efficiency.

### 1.2 Multiplication in Machine Learning

We briefly discuss the importance of matrix-vector multiplication for learning. Let  $\mathbf{x}_1, \dots, \mathbf{x}_N \in \mathbb{R}^d$  be  $N$  data points with corresponding labels  $y_1, \dots, y_N \in \mathcal{Y}$  and let  $X \in \mathbb{R}^{N \times d}$  be the feature matrix that stores  $\mathbf{x}_i$  as its  $i^{\text{th}}$  row. Matrix-vector multiplication operations abound in all phases of supervised and unsupervised learning: basic preprocessing that computes normalizing factors of the form  $X^T \mathbf{1}$  (or  $X \mathbf{1}$ ) for every feature (or data point); screening rules that use  $|X^T y|$  (when  $\mathcal{Y} \subset \mathbb{R}$ ) to exclude uninformative features (Tibs., 2010); or predictions of the form  $f(Xw)$  where  $w$  is a learned vector of weights.

Multiplication is also essential for many of the optimization techniques that lie at the core of these learning algo-

rithms. A variety of learning problems can be expressed as optimization problems of the form

$$\underset{w \in \mathbb{R}^d, \beta \in \mathbb{R}^p}{\text{minimize}} L_y(Xw, \beta) + \lambda R(w) \quad (1)$$

where  $w, \beta$  are the learning parameters,  $L_y$  is a loss function that encodes the  $y_i$  labels (if the problem is supervised), and  $R$  is a regularization penalty. It is important to remember that this framework captures a number of *unsupervised* learning problems as well, such as Principle Component Analysis, which is useful directly and as a preprocessing step for clustering, deep learning, and other techniques (Hastie, 2001). Any (sub)gradient<sup>3</sup> of (1) with respect to  $w$  is given by

$$g_w \in X^T \partial_{Xw} L(Xw, \beta) + \lambda \partial_w R(w). \quad (2)$$

where  $\partial_z f(z)$  is the subdifferential of  $f$  with respect to  $z$ .

Since every (sub)gradient descent method (Parikh, 2013) or accelerated variant critically relies on  $g_w$  as a search direction, computing  $Xw$  and then  $X^T[\partial_{Xw} L(Xw, \beta)]$  is essential and often the most costly part of the optimization. A number of other popular large-scale optimization methods also reduce to multiplying  $X$  repeatedly. These include Krylov subspace algorithms such as the conjugate gradient method, and various quasi-Newton methods including BFGS and its limited memory variant (Nocedal, 2006).

### 1.3 Background and Notation

Let  $\Sigma$  be a finite vocabulary with a strict total ordering  $\prec$  over its elements. A *document*  $D = x_1 x_2 \dots x_n$  of length  $n$  is a list of  $n$  characters drawn from  $\Sigma$  and an  $N$ -gram is any string of characters drawn from  $\Sigma$ . We will refer to each of the  $n$  suffixes in  $D$  via  $D[i] = x_i x_{i+1} \dots x_n$ . We denote the set of all substrings in  $D$  by  $D^* = \{x_i \dots x_{i+k} \mid 1 \leq i \leq n, 0 \leq k \leq n - i\}$  and the set of all substrings in a document corpus of  $N$  documents  $\mathcal{C} = \{D_1, \dots, D_N\}$  as  $\mathcal{C}^* = \bigcup_{i=1}^N D_i^*$ .

Given a subset  $\mathcal{S} \subseteq \mathcal{C}^*$  of the set of substrings in (any of) the documents, entry  $X_{is}$  of the  $N$ -gram matrix  $X \in \mathbb{Z}_+^{N \times |\mathcal{S}|}$  counts how many times substring  $s \in \mathcal{S}$  appears in document  $D_i$ . We use  $M_i$  to indicate the  $i^{\text{th}}$  column of matrix  $M$ ; when each column pertains to a specific mathematical object, such as an  $N$ -gram or tree node, we may use that object as an index (to avoid imposing a particular ordering over the objects). We will *always* take  $X$  to be an  $N$ -gram matrix for an implicitly given corpus.

A *compact tree*  $\mathcal{T} = (V, E)$  is a tree with nodes  $V$  and edges  $E$  where every internal node is required to have at least 2 children. This ensures that if  $\mathcal{T}$  has  $n$  leaves, then there are at most  $n - 1$  internal nodes. We use  $\text{ch}(v) \subset V$  and  $\text{p}(v) \in V$  to denote the children and parent of  $v \in V$ ,

<sup>3</sup>To handle non-differentiable objectives, see (Parikh, 2013).

respectively. The root node is given by  $\text{root}(\mathcal{T})$ , the depth of any node  $v \in V$  is  $d(v)$  (with  $d(\text{root}(\mathcal{T})) = 1$ ), and  $\text{depth}(T)$  is the maximum depth of any node in  $V$ . Finally, a *branch* of  $\mathcal{T}$  is a path starting at the root and ending at a leaf; we will use the terminal leaf to identify branches. We will also be concerned with subtrees  $\hat{\mathcal{T}} = (\hat{V}, \hat{E})$  of  $\mathcal{T}$  which contain a subset  $\hat{V} \subset V$  of its nodes. We allow the new edge set  $\hat{E}$  to be arbitrary and add a second argument to  $\text{ch}(v, \hat{E})$  and  $\text{p}(v, \hat{E})$  to indicate that parent/child relationships are taken with respect to this new edge set.

### 1.3.1 Suffix Trees

Given a document  $D = x_1x_2\dots x_n$  whose characters belong to an alphabet  $\Sigma$ , the *suffix tree*  $\mathcal{T}_D = (V, E)$  for  $D$  is a compact tree with  $n$  leaves, each of which corresponds to a distinct suffix of  $D$  and is numbered according to the starting position of the suffix  $1, \dots, n$ . The edges along branch  $i$  are labelled with non-empty substrings that partition  $D[i]$ : suffix  $D[i]$  can be recovered by concatenating the edge labels from the root to leaf  $i$ . Let  $l(e)$  for  $e \in E$  be the label of edge  $e$  and define the *node character*  $c(v)$  of any non-root node  $v \in V$  to be the first character of  $l(\text{p}(v), v)$ . The nodes of  $\mathcal{T}_D$  are constrained so that siblings may not have the same node character and are ordered according to the  $<$  relation on these characters. These constraints ensure that every node has at most  $|\Sigma|$  children and they allow for well-defined traversals of  $\mathcal{T}_D$ . Moreover, every substring  $s \in D^*$  is represented by a unique path in  $\mathcal{T}_D$  that starts at the root node and terminates in — possibly the middle of — an edge. Similarly to suffixes,  $s$  equals the concatenation of all characters encountered along edges from the root to the path's terminus (only a prefix of the final edge will be concatenated if the path ends in the middle of an edge).

Remarkably,  $\mathcal{T}_D$  can be constructed in  $O(n)$  time (Gusfield, 1997) and has  $n$  leaves and at most  $n - 1$  internal nodes, yet it represents all  $O(n^2)$  distinct substrings of  $D$ . This is possible because any substrings whose path representation in  $\mathcal{T}$  ends at the same edge belong to the same *equivalence class*. In particular, for  $v \in V \setminus \{\text{root}(\mathcal{T}_D)\}$  suppose that edge  $(\text{p}(v), v)$  has a label  $t = x_i \dots x_{i+k}$  and let  $s$  be the string obtained by concatenating the edge labels on the path from  $\text{root}(\mathcal{T}_D)$  down to  $v$ . Then the strings  $S(v) = \{sx_i, sx_{i+1}, \dots, sx_{i+k}\}$  belong to the same equivalence class because they occur in the same locations, i.e. if  $sx_i$  starts at location  $l$  in  $D$ , then so do all members of  $S(v)$ . For example, in the string "xaxaba" the substrings "x" and "xa" belong to the same equivalence class.

The *generalized suffix tree*  $\mathcal{T}_C$  for a document corpus  $C$  of  $n$  words compactly represents the set of all substrings in  $C^*$  and has  $n$  leaves pertaining to every suffix of every document in  $C$ . Leaves are also annotated with the document they belong to and  $\mathcal{T}_C$  inherits all of the linear-time storage and computational guarantees of the regular suffix tree (with respect to the corpus length  $n$ ).

### 1.3.2 Tree Traversals and Storage

The majority of algorithms in this paper can be expressed as a bottom-up or top-down traversal of a tree  $T = (V, E)$  (typically the suffix tree or one of its subtrees) in which information is only exchanged between a parent and its children. Given a fixed ordering of  $V$ , the necessary information for a traversal is the topology of  $T$ , i.e. its parent-child relationships, as well as any node annotations necessary for the computation. We use two formats which efficiently store this information and make traversals easy: the *breadth-first format* (BFF) and *preorder depth-first format* (DFF). In both cases we distinguish between the internal nodes and leaves of  $T$  and divide them into their respective sets  $I \cup L = V$ . In the BFF we order the nodes of  $I$  according to their *breadth-first traversal* whereas in the DFF we order the nodes of  $I$  according to their *preorder depth first traversal*; both formats assign indices  $[0, \dots, |I|]$  to the nodes in  $I$ . Note that for these traversals to be well defined we assume that the children of each node are ordered in some (arbitrary) but fixed manner. Next, the leaves of  $T$ , i.e.  $L$ , are assigned indices  $[|I|, \dots, |V|]$  so that if  $u, v \in L$  and  $p(u)$  comes before  $p(v)$  — note that both parents must be in  $I$  — then  $u$  comes before  $v$ . This ordering ensures that leaves are ordered into contiguous blocks with respect to their parent and that the blocks are in the same order as  $I$ .

A pair of arrays  $(\text{ch}^I, \text{ch}^L)$ , each of size  $|I|$ , capture the topology of  $T$ : for all  $v \in I$ ,  $\text{ch}_v^I = |\text{ch}(v) \cap I|$  stores the number of internal children of  $v$  and  $\text{ch}_v^L = |\text{ch}(v) \cap L|$  stores the number of leaf children of  $v$ . The number of bits needed to store this topology is

$$|I|(\lceil \log_2 U(I) \rceil + \lceil \log_2 U(L) \rceil) \quad (3)$$

where  $U(I), U(L)$  are the largest values in  $\text{ch}^I, \text{ch}^L$  respectively, i.e. the largest number of internal/leaf children for any node. Given node annotations in the same order as the BFF or DFF, top-down/bottom-up traversals are easy to perform by a linear sweep of the annotations and  $\text{ch}^I, \text{ch}^L$  arrays. All memory access is sequential and can be performed efficiently by standard (i.e. desktop) memory and processors.

A speed/memory tradeoff exists for the two formats. The amount of random access memory necessary for a traversal is proportional to the *depth* of  $T$  for DFF versus the *width* of  $T$  for the BFF. As we discuss in section 4, the former is likely to be smaller than the latter for our purposes. The space savings of the DFF are achieved by maintaining a stack of active nodes pertaining to the current branch being processed. The additional logic required for this book-keeping makes the DFF slightly slower than the BFF for the traversal. As such, the DFF is useful for more complicated computations in which the amount of information stored per node may be large, whereas the BFF is useful for simple computations that will be performed many times.

## 2 Fast Multiplication

This section presents our fast multiplication algorithm. Let  $\mathcal{T}_C = (V, E)$  be the suffix tree for a document corpus  $\mathcal{C} = \{D_1, \dots, D_N\}$  and let  $X$  be an  $N$ -gram matrix containing a column for every  $s \in \mathcal{S} \subseteq \mathcal{C}^*$ , i.e. the  $N$ -grams we are interested in. In order to uncover the necessary algebraic structure for our algorithm we must first remove redundant columns in  $X$ . As observed in (Zhang, 2006), redundant columns occur whenever strings in  $\mathcal{S}$  belong to the same equivalence class. This implies the following lemma:

**Lemma 1.** *For any  $v \in V$ , any  $s, s' \in \mathcal{S} \cap S(v)$  have the same distribution among the documents in  $\mathcal{C}$  so  $X_s = X_{s'}$ .*

We remove this redundancy by working with the *node matrix*  $\mathcal{X} \in \mathbb{Z}_+^{N \times M}$ , a submatrix of  $X$  that contains a single column for the  $M$  equivalence classes present in  $\mathcal{S}$ . Formally, node  $v \in V$  is present in  $X$  if  $S(v) \cap \mathcal{S} \neq \emptyset$  and we define  $\mathcal{V} \subset V$  to be the set of all nodes present in  $X$ . Column  $\mathcal{X}_v$  for  $v \in \mathcal{V}$  is obtained by picking an arbitrary  $s \in S(v) \cap \mathcal{S}$  and setting  $\mathcal{X}_v = X_s$ . We can also reconstruct  $X$  from  $\mathcal{X}$  by replicating column  $\mathcal{X}_v$   $|S(v) \cap \mathcal{S}|$  times; this underscores the inefficiency in the  $N$ -gram matrix.

### 2.1 Linear Dependencies in the Node Matrix

We are now ready to show how the topology of  $\mathcal{T}_C$  determines the linear dependencies among the columns of  $\mathcal{X}$ . Central to our analysis is the lemma below, which shows that the document frequency of any node is determined entirely by the leaves of its subtree:

**Lemma 2.** *The number of times node  $v \in V \setminus \{\text{root}(\mathcal{T}_C)\}$  appears in document  $D_i \in \mathcal{C}$  equals the number of leaves that belong to  $D_i$  in the subtree rooted at  $v$ .*

The simplest case occurs when  $\mathcal{V} = V \setminus \{\text{root}(\mathcal{T}_C)\}$ , i.e. every node in  $\mathcal{T}_C$  (except for the root) has a corresponding column in  $\mathcal{X}$ . In this case lemma 2 directly establishes a recursive definition for the columns of  $\mathcal{X}$ :

$$\mathcal{X}_v = \begin{cases} \mathbf{e}_{\text{doc}(v)}^N & \text{if } v \text{ is a leaf} \\ \sum_{u \in \text{ch}(v)} \mathcal{X}_u & \text{otherwise.} \end{cases} \quad (4)$$

Here  $\mathbf{e}_i^N$  is the  $i^{\text{th}}$  canonical basis vector for  $\mathbb{R}^N$  and  $\text{doc}(v)$  indicates the document index leaf  $v$  is labelled with. Importantly, (4) shows that the column corresponding to any internal node can be expressed as a simple linear combination of the columns of its children. This basic property lies at the core of our fast multiplication algorithm.

We now show how to apply the reasoning behind (4) to the more general case when  $\mathcal{V}$  is an arbitrary subset of  $V$ , i.e. a node's children may be partly missing. Define  $\mathcal{T}_C(\mathcal{V}) = (\hat{V}, \hat{E})$ , the *restriction* of  $\mathcal{T}_C$  to  $\mathcal{V}$ , to be a tree

with nodes  $\hat{V} = \mathcal{V} \cup \{\text{root}(\mathcal{T}_C)\}$ . In addition, for any  $v \in V \setminus \{\text{root}(\mathcal{T}_C)\}$  let  $\text{la}(v, \hat{V}) \in \hat{V}$  be the closest *proper ancestor* of  $v$  in  $\mathcal{T}_C$  that is also in  $\hat{V}$ ; since  $\text{root}(\mathcal{T}_C) \in \hat{V}$ , this mapping is always well defined. The edge set  $\hat{E}$  preserves the ancestor relationships among the nodes in  $\hat{V}$ : every  $v \in \mathcal{V}$  is connected to  $\text{la}(v, \hat{V})$  as a child. An inductive argument shows that if  $u, v \in \hat{V}$ , then  $u$  is an ancestor of  $v$  in  $\mathcal{T}_C$  if and only if  $u$  is also an ancestor of  $v$  in  $\mathcal{T}_C(\mathcal{V})$ .

Associated with  $\mathcal{T}_C(\mathcal{V})$  is a matrix  $\Phi \in \mathbb{Z}_+^{N \times |\mathcal{V}|}$  that subsumes the role of leaf document labels.  $\Phi$  contains a column for every node  $v \in \mathcal{V}$  and accounts for all of the leaves in  $\mathcal{T}_C$ . When  $v$  is a leaf in  $\mathcal{T}_C$  and  $v$  is included in  $\mathcal{V}$  we set  $\Phi_v = \mathbf{e}_{\text{doc}(v)}^N$ . Otherwise,  $v$  is accounted for in  $\Phi_{\text{la}(v, \hat{V})}$ , the column pertaining to  $v$ 's closest ancestor in  $\mathcal{V}$ . In particular, if  $u \in \mathcal{V}$  is *not* a leaf in  $\mathcal{T}_C$ , then

$$\Phi_u = \sum_{\substack{v \in \text{leaves}(\mathcal{T}_C) \setminus \mathcal{V} \\ \text{la}(v, \hat{V}) = u}} \mathbf{e}_{\text{doc}(v)}^N. \quad (5)$$

This bookkeeping allows us to relate the columns of  $\mathcal{X}$  when  $\mathcal{V}$  is any subset of  $V$ :

**Theorem 1.** *The columns of the node matrix  $\mathcal{X}$  for  $\mathcal{V} \subseteq V \setminus \{\text{root}(\mathcal{T}_C)\}$  are given recursively by*

$$\mathcal{X}_v = \Phi_v + \sum_{u \in \text{ch}(v; \hat{E})} \mathcal{X}_u$$

where  $\Phi$  and  $\mathcal{T}_C(\mathcal{V}) = (\hat{V}, \hat{E})$  are defined above.

This theorem shows that  $\mathcal{X}_v$  is a simple linear combination of the columns of its children in  $\mathcal{T}_C(\mathcal{V})$  plus a correction term in  $\Phi$ . We utilize this structure below to give a fast matrix-vector multiplication algorithm for node matrices.

### 2.2 Fast Multiplication Algorithm

A simple application of Theorem 1 shows that the matrix-vector product  $\mathcal{X}w$  for  $w \in \mathbb{R}^{|\mathcal{V}|}$  can be obtained by recursively collecting entries of  $w$  into a vector  $\beta \in \mathbb{R}^{|\mathcal{V}|}$ :

$$\beta_v = w_v + \beta_{\text{p}(v; \hat{E})} \quad (6a)$$

$$\mathcal{X}w = \Phi\beta \quad (6b)$$

Here we use the convention  $\beta_{\text{root}(\mathcal{T}_C(\mathcal{V}))} = 0$ . The transposed operation  $\mathcal{X}^T y$  for  $y \in \mathbb{R}^N$  can also be written recursively by expressing each entry as

$$(\mathcal{X}^T y)_v = \Phi_v^T y + \sum_{u \in \text{ch}(v; \hat{E})} (\mathcal{X}^T y)_u. \quad (7)$$

Equations (6-7) lead to the following theorem, for which we provide a proof sketch:

<sup>4</sup>Note that  $\mathcal{V}$  never includes the root node.

**Theorem 2.** *Let  $\mathcal{C}$  be a document corpus of  $n$  words and let  $\mathcal{X}$  be any node matrix derived from this corpus. Then  $\mathcal{X}$  requires  $O(n)$  memory to store. Multiplying a vector with  $\mathcal{X}$  or  $\mathcal{X}^T$  requires  $O(n)$  operations.*

*Proof.* Vector  $\beta$  in equation (6) can be computed in  $O(|\mathcal{V}|) \in O(n)$  operations by a top-down traversal that computes each of its entries in constant time. The matrix  $\Phi$  is a sparse matrix with *at most* one entry per leaf of the suffix tree  $\mathcal{T}_{\mathcal{C}}$ , i.e. at most  $n$  entries. It follows that the product  $\Phi\beta$  requires  $O(n)$  operations which proves the theorem for multiplication with  $\mathcal{X}$ . The transposed case follows similarly by noting that we must compute a matrix-vector product with  $\Phi^T$  and perform a bottom-up traversal that performs constant time operations for every node in  $\mathcal{V}$ .  $\square$

### 2.2.1 Efficiency Gains

We use naïve multiplication to mean sparse matrix-vector multiplication in what follows. The supplementary material discusses examples which show that naïve multiplication with the  $N$ -gram matrix  $X$  can be asymptotically slower than naïve multiplication with  $\mathcal{X}$ , which in turn can be asymptotically slower than multiplication with our recursive algorithm. These examples establish the following complexity separation result:

**Theorem 3.** *There exists documents of  $n$  words for which*

1. *The all  $N$ -grams matrix  $X$  requires  $\Theta(n^2)$  storage and operations to multiply naïvely.*
2. *The all  $N$ -grams node matrix  $\mathcal{X}$  requires  $\Theta(n\sqrt{n})$  storage and operations to multiply naïvely.*
3. *In all cases recursive multiplication of the node matrix requires  $O(n)$  storage and operations.*

### 2.3 Matrix Data Structure

The fast multiplication algorithm can be performed directly on the suffix tree derived from  $\mathcal{C}$ , but it is faster to use a dedicated data structure optimized for the algorithm’s memory access patterns. The breadth-first multiplication tree (BFMT) stores the topology of  $\mathcal{T}_{\mathcal{C}}(\mathcal{V})$  in the BFF (discussed in section 1.3.2) and the frequency information in  $\Phi$  as a sparse matrix in a modified compressed sparse column (csc) format (see the supplementary material) whose columns are ordered according to the order of the BFF. We chose this format because executing equations (6) and (7) requires a simple linear sweep of the BFMT. We expect that the vectors being multiplied can be stored in memory and therefore opted for the speed afforded by the BFF instead of the memory savings of the DFF.

The total number of bits necessary to store the BFMT is given by equation (3) along with the total number of bits

necessary to store  $\Phi$ , which is

$$|\mathcal{V}| \lceil \log_2 U^\Phi \rceil + \text{nz}(\lceil \log_2 M \rceil + \lceil \log_2 N \rceil). \quad (8)$$

Here  $U^\Phi = \max_{v \in \mathcal{V}} \|\Phi_v\|_0$  is the largest number of non-zero elements in a column of  $\Phi$ ,  $\text{nz}$  is the total number of non-zero elements in  $\Phi$ , and  $M$  is the largest entry in  $\Phi$ . It is easy to verify that  $|\mathcal{I}| \leq |\mathcal{V}| \leq \text{nz} \leq n$  and the term involving  $\text{nz}$  typically dominates the memory requirements.

## 3 Common Usage Patterns

We now discuss how several common machine learning scenarios can be adapted to use our representation of the node matrix or preferably, to treat multiplication with  $\mathcal{X}$  as a black-box routine. The most straightforward use case is to replace the original  $N$ -gram matrix with the more succinct node matrix. Moreover, mean centering and column normalization can be performed *implicitly*, without modifying  $\mathcal{X}$ , by premultiplying and adding a correction term:

$$((\mathcal{X} - \mathbf{1}\mu^T)\Sigma)w = \mathcal{X}(\Sigma w) - (\mu^T \Sigma w)\mathbf{1}$$

Here  $\mu$  is a vector of column means and  $\Sigma$  is a diagonal matrix of column normalizing factors. Analogous formulas exist for row centering and normalization.

### 3.1 Problem Reformulation

A variety of optimization problems can also be reformulated so that they are *equivalent* to using the original  $N$ -gram matrix. A simple example of such a conversion comes from using ridge regression to model label  $y_i \in \mathbb{R}$  based on the  $i^{\text{th}}$  row of the  $N$ -gram matrix  $X$ . We wish to solve

$$\underset{w \in \mathbb{R}^d}{\text{minimize}} \frac{1}{2} \|y - Xw\|_2^2 + \frac{\lambda}{2} \|w\|_2^2. \quad (9)$$

It is easy to show that if  $\lambda > 0$  and  $N$ -grams  $s, t$  belong to the same equivalence class, then  $w_s = w_t$ . We can simulate the effect of these duplicated variables by collecting terms. Let  $\mathcal{S}$  be the set of  $N$ -grams present in  $X$ ,  $\mathcal{V}$  the set of suffix tree nodes present in  $X$ , and define  $\mathcal{S}(v) = \mathcal{S}(v) \cap \mathcal{S}$  for brevity. For all  $v \in \mathcal{V}$  let  $z_v = |\mathcal{S}(v)|w_s$  for some  $s \in \mathcal{S}(v)$ . Then  $\sum_{s \in \mathcal{S}(v)} X_s w_s = \mathcal{X}_v z_v$  and  $\sum_{s \in \mathcal{S}(v)} w_s^2 = |\mathcal{S}(v)|^{-1} z_v^2$  so problem (9) is equivalent to a *smaller* weighted ridge regression using  $\mathcal{X}$ :

$$\underset{z \in \mathbb{R}^{|\mathcal{V}|}}{\text{minimize}} \frac{1}{2} \|y - \mathcal{X}z\|_2^2 + \frac{\lambda}{2} \sum_{v \in \mathcal{V}} \frac{z_v^2}{|\mathcal{S}(v)|}. \quad (10)$$

Note that this also shows that representations using the  $N$ -gram matrix downweight the ridge penalty of each equivalence class in proportion to its size.

We can characterize the set of optimization problems that have an equivalent problem where the  $N$ -gram matrix can

be replaced with the node matrix. Define a partition  $\mathcal{J}$  of the integer set  $\{1, \dots, d\}$  to be a set of  $m$  integral intervals  $\zeta_k = \{i, \dots, j\}$  such that  $\bigcup_{k=1}^m \zeta_k = \{1, \dots, d\}$  and  $\zeta_k \cap \zeta_j = \emptyset$  if  $k \neq j$ . A function  $f : \mathbb{R}^d \rightarrow \mathbb{R}^p$  is *permutation invariant* with respect to  $\mathcal{J}$  (abbrev.  $\mathcal{J}$ -PI) if for all  $x \in \mathbb{R}^d$ ,  $f(x) = f(\pi[x])$  where  $\pi : \mathbb{R}^d \rightarrow \mathbb{R}^d$  is any permutation that only permutes indices within the same interval  $\zeta_k \in \mathcal{J}$ . For our purposes it is important to note that  $L_p$ -norms are  $\mathcal{J}$ -PI as are affine functions  $Ax + b$  whenever columns  $A_i = A_j \forall i, j \in \zeta_k, \forall \zeta_k \in \mathcal{J}$ . It is straightforward to show that if  $f, g : \mathbb{R}^d \rightarrow \mathbb{R}^p$  are both  $\mathcal{J}$ -PI and  $c : \mathbb{R}^p \rightarrow \mathbb{R}^q$  then  $f(x) + g(x)$  and  $c(f(x))$  are also  $\mathcal{J}$ -PI.

We prove the following theorem in the supplementary material to connect permutation invariance to optimization:

**Theorem 4.** *Let  $f : \mathbb{R}^d \rightarrow \mathbb{R}$  be any convex function that is  $\mathcal{J}$ -PI where  $m = |\mathcal{J}|$ . Then there exists a convex function  $g : \mathbb{R}^m \rightarrow \mathbb{R}$  over  $m$  variables such that the problem  $\min_{x \in \mathbb{R}^d} f(x)$  is equivalent to  $\min_{z \in \mathbb{R}^m} g(z)$ . If  $z^*$  is optimal for the second problem, then  $x_i = z_k^* \forall i \in \zeta_k, \forall \zeta_k \in \mathcal{J}$  is optimal for the first problem.*

This theorem establishes that any convex loss of the form  $L(Xw, b)$ ; e.g. SVM, logistic regression, least squares; added to any  $L_p$  norm, e.g.  $L_2$  ridge or  $L_1$  lasso penalty, can be simplified to an equivalent learning problem that uses the node matrix instead of the  $N$ -gram matrix.

### 3.2 Holding Out Data

Oftentimes the document corpus is organized into  $T$  (possibly overlapping) integral sets  $\mathcal{Q}_1, \dots, \mathcal{Q}_T$  indexing the documents. For instance, splitting documents into training and testing sets yields  $T = 2$  index sets, and further subdividing the training set for  $K$ -fold crossvalidation introduces  $2K$  additional sets (indicating the hold out and training data for each split). In this case we are not interested in multiplying all of  $\mathcal{X}$ , but only the submatrix whose rows' indices are in the given  $\mathcal{Q}_i$ . This matrix-vector product can be computed by calling the recursive multiplication algorithm with the topology information in  $\mathcal{T}_C(\mathcal{V})$  (derived from the full corpus) and with the submatrix of  $\Phi$  whose rows' indices are in  $\mathcal{Q}_i$ . Also note that if only a subset of the documents will ever be used for training, we can ignore any nodes in  $\mathcal{T}_C$  that do not appear in the training set since they (should) be ignored by any learning algorithm; we discuss this further in section 4.2.

## 4 Preprocessing

We use an intermediary data structure, the depth-first preprocessing tree (DFPT), to output the BFMT. The DFPT is computed from a corpus  $\mathcal{C}$  and stores the *minimal* information in  $\mathcal{T}_C = (V, E)$  necessary to produce any BFMT and to prune the nodes in  $V$ . It can be computed once and used to store  $\mathcal{C}$  in a format that is amenable for arbitrary machine

learning tasks. Given a new learning problem the DFPT proceeds in two stages: 1) it identifies useful  $N$ -grams in  $V$  and calculates relevant column normalizations, and 2) it emits a BFMT tailored to that task. Construction of the DFPT, as well as its processing stages, requires  $O(n)$  time and memory with respect to the corpus length  $n$ .

As suggested by its name, the DFPT stores the topology of  $\mathcal{T}_C$  in DFF, its leaf-document annotations, and if filtering by  $N$ -gram length, the edge label length for each internal node of  $V$ . Its processing stages are a sequence of top-down/bottom-up traversals of  $\mathcal{T}_C$  that are individually more sophisticated than those required by our multiplication algorithm, so we opted for the memory savings afforded by the DFF. Indeed,  $\text{depth}(\mathcal{T}_C)$  is bounded by the length of the longest document in  $\mathcal{C}$  while the tree width is bounded by the corpus length; the memory savings of the DFF over the BFF are substantial. Importantly, the traversals stream through the DFPT so it is reasonable to operate on it via external memory, e.g. a hard drive, if memory is limited.

In detail, the DFPT requires  $2|I| \lceil \log_2 |\Sigma| \rceil + n \lceil \log_2 N \rceil$  bits to store the topology and leaf-document annotations, where  $I$  is the set of internal nodes of  $V$ ,  $N$  the number of documents in  $\mathcal{C}$ , and  $\Sigma$  the alphabet. For reference storing  $\mathcal{C}$  requires  $n \lceil \log_2 |\Sigma| \rceil + N \lceil \log_2 n \rceil$  bits and  $|I| < n$ . An additional  $|I| \lceil \log_2 \frac{n}{N} \rceil$  bits are used to store edge labels lengths, but this information is only necessary when pruning by maximum  $N$ -gram length.

### 4.1 Computing the Depth-First Suffix Tree

Computing the DFPT from  $\mathcal{C}$  represents the least memory efficient part of our framework as we first compute a suffix array (SA) (Gusfield, 1997) from the text and then convert the SA into the DFPT. The process requires  $3n \lceil \log_2 n \rceil + n \lceil \log_2 (|\Sigma| + N) \rceil$  bits and  $O(n)$  time. We emphasize that our framework is completely modular so the DFPT only needs to be computed once. We leave it as an open problem to determine if a more memory efficient algorithm exists that directly computes the DFPT.

Recalling that each leaf of  $\mathcal{T}_C$  is numbered according to the suffix it corresponds to, the SA is a permutation of the integers  $[0, \dots, n)$  that stores the leaves of  $\mathcal{T}_C$  in a *pre-order depth-first traversal*. We use an SA rather than a suffix tree because the former typically requires 4 times less memory than a suffix tree and can also be constructed in  $O(n)$  time and memory. We use the implementation of (Mori, 2015), which requires  $m = 3n \lceil \log_2 n \rceil + n \lceil \log_2 (|\Sigma| + N) \rceil$  bits to construct the SA, where the second term corresponds to a modified copy of  $\mathcal{C}$ . This was the most memory efficient linear-time suffix array construction algorithm we could find; asymptotically slower but more memory efficient algorithms may be preferable for DNA sequence data.

The details of how we compute the DFPT from a suffix

array are rather involved and will be discussed in an extended version of this paper. The framework of (Kasai, 2001) is instrumental in our conversion as it allows us to simulate a *post-order depth-first traversal* of  $\mathcal{T}_C$  using the SA. By carefully managing memory and off-loading unused information to external storage, each step of the conversion requires at most  $m - n\lceil\log_2 n\rceil$  bits to be stored in main memory at any time. The *total* memory requirements, including storing the DFPT while it is constructed, never exceed the maximum of  $m - n\lceil\log_2 n\rceil$  and  $2n\lceil\log_2 n\rceil + (n + |I|)\lceil\log_2 N\rceil$  bits; both are less than  $m$ .

## 4.2 Filtering and Normalizations

The first stage of the DFPT’s processing determines which nodes in  $\mathcal{T}_C$  should be present in the final BFMT. It also computes any required normalizations, such as the column mean or norm, of the node matrix  $\mathcal{X}$  the BFMT represents. We assume that only the internal nodes  $I \subset V$  of  $\mathcal{T}_C$  will ever be used; each leaf appears in only a single document and is unlikely to carry useful information. We model the screening process as a sequence of filters that are applied to  $I$ : associated with  $I$  is a boolean array  $b \in \{0, 1\}^{|I|}$  where  $b_v = 1$  indicates that node  $v \in I$  is useful and  $b_v = 1 \forall v \in I$  initially. Each filter takes as input the DFPT and  $b$ , and updates  $b$  (in place) with its own criteria. All of our filters are memory efficient and only need to store  $|I| + O(\text{depth}(\mathcal{T}_C))$  bits in memory as the BFMT can reasonably be streamed from slower external storage. With the exception of the unique document filter, all of the filters listed below run in  $O(n)$  time:

**$N$ -gram length:** removes nodes whose shortest corresponding  $N$ -gram is longer than a given threshold.

**Training set:** removes nodes that do not appear in any documents designated as the training set.

**Unique document frequency:** removes nodes that do not appear in at least some number of *distinct* documents. We use an algorithm given in (Paskov, 2015) which runs in  $O(n\alpha^{-1}(n))$  time, where  $\alpha^{-1}$  is the inverse Ackermann function ( $\alpha^{-1}(10^{80}) = 4$ ) and is essentially linear-time. A  $O(n)$  algorithm (Gusfield, 1997) is possible, but it requires complicated pre-processing and an additional  $n\lceil\log_2 n\rceil$  bits of memory.

**Strong rules:** given *mean centered* document labels  $y \in \mathbb{R}^N$ , removes all nodes  $v$  for which  $|\mathcal{X}_v^T y| < \lambda$  for a threshold  $\lambda$ . This implements the strong rules of (Tibs., 2010) and can be applied to a subset of the documents  $\mathcal{I}_r \subset \{1, \dots, N\}$  (e.g. training data) by mean centering only  $y_{\mathcal{I}_r}$  and setting  $y_i = 0$  for all  $i \notin \mathcal{I}_r$ . Column normalizations are achieved by checking  $\eta_v^{-1}|\mathcal{X}_v^T y| < \lambda$ , where  $\eta_v^{-1}$  is the normalization for column  $v$ . This filter essentially multiplies  $\mathcal{X}^T y$  using the DFPT and the normalization can be computed on the fly (see discussion below).

Once we know which nodes will be used in the BFMT we typically require the column mean  $\mu = \frac{1}{N}\mathcal{X}^T \mathbf{1}$  and some kind of normalization  $\eta_v^{-1}$  for each column of  $\mathcal{X}$ . Noting that all entries of  $\mathcal{X}$  are non-negative, the  $L_1$ -norm of each column is  $\eta = \mathcal{X}^T \mathbf{1}$ . Each of these quantities is a matrix-vector multiply that we perform using the DFPT. These quantities can be specialized to training data by setting appropriate entries of the  $\mathbf{1}$  vector to 0. We can also compute the  $L_2$ -norm of each column of  $\mathcal{X}$  or the  $L_1/L_2$ -norm of each column of  $\mathcal{X} - \mathbf{1}\mu^T$ , the mean centered node matrix. These normalizations however require  $O(N|I|)$  time and  $O(N\text{depth}(\mathcal{T}_C))$  memory; the space savings of the DFF are critical for the memory bound. These running times are tolerable if performed only once, especially on the short and wide trees that tend to occur with natural language.

## 4.3 Producing the Matrix Structure

The final stage in our pipeline produces the BFMT using the DFPT and filter  $b$ . The following lemma follows from the definitions of breadth-first and depth-first traversals and is essential for easy conversion between the two formats:

**Lemma 3.** *Given a tree  $T = (V, E)$ , let  $\beta$  be an (ordered) list of the nodes in  $V$  in breadth-first order and define  $\delta$  to be a list of  $V$  in depth-first preorder. Define  $\beta(d)$  and  $\delta(d)$  to be the (sub) lists of  $\beta$  and  $\delta$  respectively containing only nodes at depth  $d$ . Then  $\beta(d) = \delta(d) \forall d = 1, \dots, \text{depth}(T)$ .*

This lemma states that the breadth-first and depth-first preorder traversals list nodes in the same order if we only consider the nodes of a tree at a specific depth. We thus allocate memory for the BFMT by counting the number of nodes with  $b_v = 1$  at each depth in the DFPT. The lemma then allows us to copy the relevant nodes in the DFPT into the BFMT skeleton by maintaining a stack of size  $\text{depth}(\mathcal{T}_C)$  that keeps track of how many nodes have been written to the BFMT at each depth. The depth-first traversal also makes it is easy to determine edges by keeping track of each node’s nearest ancestor (in  $\mathcal{T}_C$ ) that is in the BFMT. The copying process streams through the DFPT and  $b$  in a single linear sweep and requires storing the BFMT and  $O(\text{depth}(\mathcal{T}_C))$  bits in memory.

## 5 Experiments

This section provides benchmarks for our multiplication algorithm and applies it to solve several large-scale sentiment analysis tasks. We implemented our framework in  $C^5$  and compiled it used the GCC compiler version 4.4.7 for an x86-64 architecture. Our reference machine uses an Intel Xeon E5-2687W processor with 8 cores running at 3.1 GHz and has 128 Gb of RAM.

<sup>5</sup>Please contact the first author for source code.



## 5.1 Memory and Timing Benchmarks

We evaluate our multiplication algorithm on three kinds of data to investigate its performance in a variety of scenarios: short natural language articles, long technical papers, and DNA sequence data. The first is the BeerAdvocate dataset (McAuley, 2013), a corpus of 1,586,088 beer reviews totalling 1 Gb of plaintext and each consisting of a median of 126 words. The second is a collection of 70,728 journal articles collected from NCBI (U.S. National Library, 2015) with a median length of 6955 words and totalling 3 Gb of plaintext<sup>6</sup>. Our third dataset is derived from the 1000 Genomes Project (1000 Genomes Project, 2008) and it consists of 6,196,151 biallelic markers, i.e. binary values, along chromosome 1 for 250 people.

Our preprocessing framework required at most 3.5 times the memory of the original datasets for the natural language data. The third dataset however presents a worst case scenario for our framework and suffix tree/arrays in general. It requires 185 megabytes to store because of its small alphabet size, yet its suffix array requires  $n \lceil \log_2 n \rceil$  bits, i.e. 31 times more memory, and several times this amount to compute. While the DFPT ameliorates this memory usage, it still requires 10 times more memory than the original data and total memory usage went up to 18 gigabytes when computing it from the suffix array.

Figure 1 compares the memory requirements of the BFMT to explicitly storing the node and  $N$ -gram matrices for all  $N$ -grams up to length  $K$  that appear in at least 2 documents. We show space requirements for our modified sparse matrix format (MSF) as well as the standard sparse format (SSF), e.g. used in Matlab. The top two graphs correspond to the natural language datasets and have similar patterns: memory usage for the explicit representations rises quickly for up to  $K = 7$  and then tapers off as overly long  $N$ -grams are unlikely to appear in multiple documents. In all cases the BFMT is superior, requiring approximately 3 times less memory than the MSF and up to 14 times less memory than its floating-point counterpart. While there is some separation between the node matrix and naïve all  $N$ -gram matrix, the gap – which is more pronounced in the journal articles – is mitigated by filtering  $N$ -grams that do not appear in multiple documents.

The third graph presents a striking difference between the representations: the BFMT requires up to 41 times less memory than the MSF node matrix and over 4,600 times less memory than the naïve  $N$ -gram matrix. The floating point counterparts for these matrices accentuate the differences by a factor of 5. Interestingly, the size of the BMFT decreases as  $K$  increases from  $10^3$  to  $10^4$ . This occurs because when  $K \geq 10^4$ , the BFMT behaves as if all  $N$ -grams are included so that all entries in the frequency matrix  $\Phi$  are

<sup>6</sup>This data was graciously made available by the Saccharomyces Genome Database at Stanford.

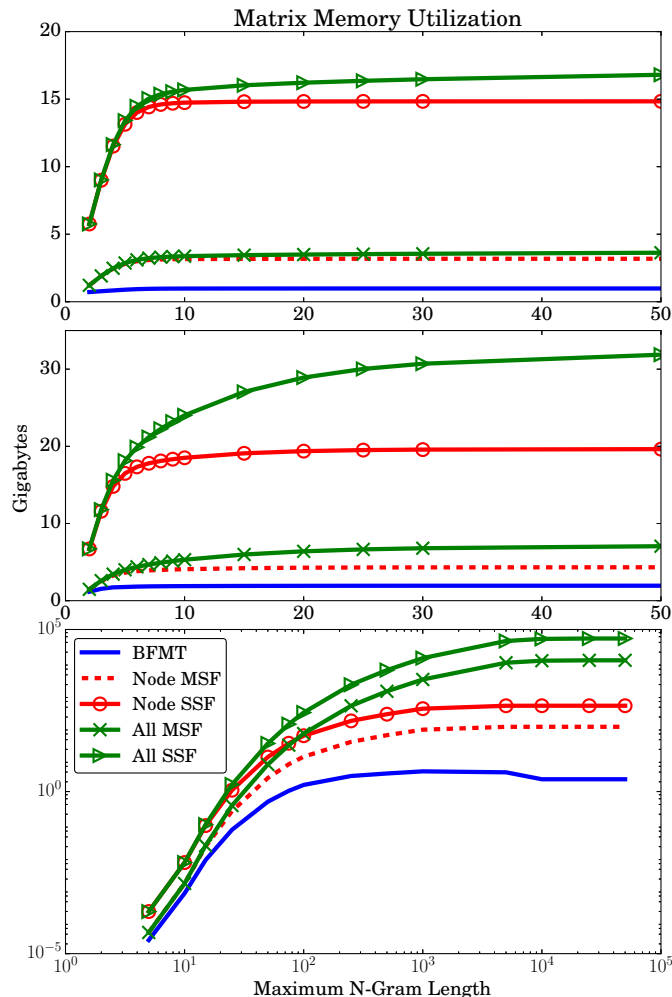


Figure 1: Memory utilization for the BFMT, node, and all  $N$ -gram matrices as a function of maximum  $N$ -gram length  $K$  on the BeerAdvocate data (top), journal data (middle) and 1000 genomes data (bottom).

0 or 1. When  $K \approx 10^3$ , most of the entries are bounded by 1, but a few large entries exist and force additional bits to be used for *all* non-zero frequencies in  $\Phi$ .

Next, figure 2 compares the average multiplication time for the BFMT to ordinary sparse multiplication with the node matrix. The top figure presents results for the BeerAdvocate data; we did not include timings for the journal data since they are essentially the same. We were unable to provide timing results for the node matrix on the DNA data because it quickly exceeded our computer’s memory. All trials were performed using a single core for fairness of comparison. The difference between the BFMT and the node matrix closely follows the memory requirement differences. This is to be expected as both multiplication routines make a single pass over the data so running time is proportional to the amount of data that must be scanned. We also note that the BFMT running time scales gracefully

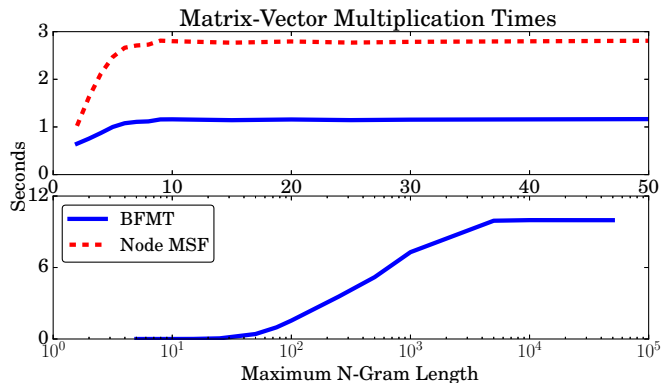


Figure 2: Average time to perform one matrix-vector multiplication with the BFMT and node matrices as a function of maximum  $N$ -gram length  $K$  on the BeerAdvocate data (top) and 1000 Genomes Data (bottom). Node matrix times are missing for the latter because it was impractical to store.

on the DNA data; time increases at a logarithmic rate with respect to  $K$  since the  $x$ -axis is logarithmic.

## 5.2 Sentiment Analysis Tasks

We applied our framework to sentiment analysis tasks on three large datasets: the BeerAdvocate dataset, a set of 6,396,350 music reviews from Amazon (McAuley, 2013) (4.6 Gb of text), and a set of 7,850,072 movie reviews also from Amazon (McAuley, 2013) (7.4 Gb of text). Each review’s sentiment is a value between 1 and 5 (indicating negative or positive) and we tried to predict this sentiment using a ridge regression model on features provided by the node matrix. Each dataset was randomly split into training, validation, and testing sets comprised of 75%, 12.5%, and 12.5% of the total data; all parameters discussed below were selected based on their validation set performance.

We solved the regression by implementing a conjugate gradient solver in C that uses our fast multiplication routine. The ridge parameter  $\lambda$  was tuned on a grid of up to 100 values. We stopped tuning once the validation error increased for 5 consecutive  $\lambda$  values and the procedure typically terminated after trying 60 values.  $N$ -grams were pruned by maximum  $N$ -gram length and were required to appear in at least 20 distinct documents – we experimented with several document frequency thresholds. We also used the strong rules to select a subset of the features for each  $\lambda$  value and used  $\alpha\lambda$  as the threshold;  $\alpha = 1$  always gave the best performance. Finally, all columns were mean centered and normalized by their  $L_2$  norm. Our framework computed this normalization in 2.5 minutes for the larger movie dataset. The largest and most time intensive feature set contained 19.4 million features and occurred for  $K = 5$  on the movie dataset. It took 26 hours to solve for and evaluate 69 lambda values while running on a *single* core. We

were able to effectively run all  $N$ -gram trials in parallel.

Table 1: Mean Squared Error for Sentiment Analysis

| K | Beer         | Music        | Movies       |
|---|--------------|--------------|--------------|
| 1 | 0.286        | 0.766        | 0.765        |
| 2 | 0.254        | 0.481        | 0.237        |
| 3 | 0.245        | 0.366        | 0.140        |
| 4 | <b>0.244</b> | 0.333        | 0.121        |
| 5 | <b>0.244</b> | <b>0.325</b> | <b>0.115</b> |

Table 1 summarizes the mean-squared error of our regression model on the testing set. All three datasets benefit from longer  $N$ -grams, but we note that the longer datasets seem to benefit more (size increases from left to right). Confounding this potential effect are peculiarities specific to the tasks and specific to BeerAdvocate versus Amazon reviews (recall that the music and movie reviews both come from the same data source). Nonetheless, it is also possible that larger datasets are better equipped to utilize long  $N$ -grams: they provide enough examples to counter the variance incurred from estimating coefficients for long, and therefore relatively infrequent,  $N$ -grams. It will be interesting to verify this potential effect with more experiments.

## 6 Conclusion

This paper shows that learning with long  $N$ -grams on large corpora is tractable because of the rich structure in  $N$ -gram matrices. We provide a framework that can be used to permanently store a document corpus in a format optimized for machine learning. Given a particular learning task, our framework finds helpful  $N$ -grams and emits a data structure that is tailored to the problem and allows for fast matrix-vector multiplication – an operation that lies at the heart of many popular learning algorithms.

Our work suggests a number of extensions. While our multiplication algorithm is single-threaded, the underlying data structure is a tree and the necessary traversals are easy to parallelize. Finding an efficient algorithm to directly compute the DFPT without using a suffix array will also be useful. Finally, our framework provides considerable computational savings for DNA sequence data over naïve representations (up to 4 orders of magnitude). We hope that our algorithm will inspire additional research into long  $N$ -gram models for this kind of data.

## Acknowledgements

In loving memory of Христо П. Георгиев (Hristo P. Georgiev). Funding provided by the Air Force Office of Scientific Research and the National Science Foundation.

## References

- [1] Mohamed Ibrahim Abouelhoda, Stefan Kurtz, and Enno Ohlebusch. Replacing suffix trees with enhanced suffix arrays. *J. of Discrete Algorithms*, 2(1):53–86, March 2004.
- [2] William Cavnar and John Trenkle. N-gram-based text categorization. In *In Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175, 1994.
- [3] Dan Gusfield. *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*. Cambridge University Press, New York, NY, USA, 1997.
- [4] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.
- [5] John Houvardas and Efstathios Stamatatos. N-gram feature selection for authorship identification. In *Proceedings of the 12th International Conference on Artificial Intelligence: Methodology, Systems, and Applications, AIMS'06*, pages 77–86, Berlin, Heidelberg, 2006. Springer-Verlag.
- [6] Juhani Kähärä and Harri Lähdesmäki. Evaluating a linear k-mer model for protein-dna interactions using high-throughput selex data. *BMC Bioinformatics*, 14(S-10):S2, 2013.
- [7] Toru Kasai, Hiroki Arimura, and Setsuo Arikawa. Efficient substring traversal with suffix arrays.
- [8] Julian McAuley and Jure Leskovec. From amateurs to connoisseurs: Modeling the evolution of user expertise through online reviews. In *Proceedings of the 22Nd International Conference on World Wide Web, WWW '13*, pages 897–908, Republic and Canton of Geneva, Switzerland, 2013. International World Wide Web Conferences Steering Committee.
- [9] Julian McAuley and Jure Leskovec. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 165–172. ACM, 2013.
- [10] Yuta Mori. sais, <https://sites.google.com/site/yuta256/sais>, 2015.
- [11] Jorge Nocedal and Stephen Wright. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer New York, 2006.
- [12] U.S. National Library of Medicine. National center for biotechnology information, 2015.
- [13] Neal Parikh and Stephen Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 2013.
- [14] Hristo Paskov, John Mitchell, and Trevor Hastie. Uniqueness counts: A nearly linear-time online algorithm for the multiple common substring problem. In *In preparation*, 2015.
- [15] Hristo Paskov, Robert West, John Mitchell, and Trevor Hastie. Compressive feature learning. In *NIPS*, 2013.
- [16] 1000 Genomes Project. 1000 genomes project, a deep catalog of human genetic variation., 2008.
- [17] Konrad Rieck and Pavel Laskov. Linear-time computation of similarity measures for sequential data. *The Journal of Machine Learning Research*, 9:23–48, 2008.
- [18] Choon Hui Teo and S. V. N. Vishwanathan. Fast and space efficient string kernels using suffix arrays. In *In Proceedings, 23rd ICMP*, pages 929–936. ACM Press, 2006.
- [19] Robert Tibshirani. Regression shrinkage and selection via the lasso. *J. R. Stat. Soc. B*, 58(1):267–288, 1996.
- [20] Robert Tibshirani, Jacob Bien, Jerome Friedman, Trevor Hastie, Noah Simon, Jonathan Taylor, and Ryan J. Tibshirani. Strong rules for discarding predictors in lasso-type problems., 2010.
- [21] SVN Vishwanathan and Alexander Johannes Smola. Fast kernels for string and tree matching. *Kernel methods in computational biology*, pages 113–130, 2004.
- [22] Sida Wang and Christopher Manning. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the ACL*, pages 90–94, 2012.
- [23] Frank Wood, Jan Gasthaus, Cédric Archambeau, Lancelot James, and Yee Whye Teh. The sequence memoizer. *Communications of the ACM*, 54(2):91–98, 2011.
- [24] Dell Zhang. Extracting key-substring-group features for text classification. In *In Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD 06)*, pages 474–483. ACM Press, 2006.

---

# A Complete Generalized Adjustment Criterion

---

|  |   |   |   |
|--|---|---|---|
| <b>Emilija Perković</b><br>Seminar for Statistics<br>ETH Zurich, Switzerland<br>perkovic@stat.math.ethz.ch | <b>Johannes Textor</b><br>Theoretical Biology & Bioinformatics<br>Utrecht University, The Netherlands<br>johannes.textor@gmx.de | <b>Markus Kalisch</b><br>Seminar for Statistics<br>ETH Zurich, Switzerland<br>kalisch@stat.math.ethz.ch | <b>Marloes H. Maathuis</b><br>Seminar for Statistics<br>ETH Zurich, Switzerland<br>maathuis@stat.math.ethz.ch |
|--|---|---|---|

## Abstract

Covariate adjustment is a widely used approach to estimate total causal effects from observational data. Several graphical criteria have been developed in recent years to identify valid covariates for adjustment from graphical causal models. These criteria can handle multiple causes, latent confounding, or partial knowledge of the causal structure; however, their diversity is confusing and some of them are only sufficient, but not necessary. In this paper, we present a criterion that is necessary and sufficient for four different classes of graphical causal models: directed acyclic graphs (DAGs), maximum ancestral graphs (MAGs), completed partially directed acyclic graphs (CPDAGs), and partial ancestral graphs (PAGs). Our criterion subsumes the existing ones and in this way unifies adjustment set construction for a large set of graph classes.

## 1 INTRODUCTION

Which covariates do we need to adjust for when estimating total causal effects from observational data? Graphical causal modeling allows to answer this question constructively, and contributed fundamental insights to the theory of adjustment in general. For instance, a simple example known as the “M-bias graph” shows that it is not always appropriate to adjust for all observed (pre-treatment) covariates (Shrier, 2008; Rubin, 2008). A few small graphs also suffice to refute the “Table 2 fallacy” (Westreich and Greenland, 2013), which is the belief that the coefficients in multiple regression models are “mutually adjusted”. Thus, causal graphs had substantial impact on theory and practice of covariate adjustment (Shrier and Platt, 2008).

The practical importance of covariate adjustment has inspired a growing body of theoretical work on graphical criteria that are sufficient and/or necessary for adjustment.

Pearl’s back-door criterion (Pearl, 1993) is probably the most well-known, and is sufficient but not necessary for adjustment in DAGs. Shpitser et al. (2012) adapted the back-door criterion to a necessary and sufficient graphical criterion for adjustment in DAGs. Others considered graph classes other than DAGs, which can represent structural uncertainty. van der Zander et al. (2014) gave necessary and sufficient graphical criteria for MAGs that allow for unobserved variables (latent confounding). Maathuis and Colombo (2015) presented a generalized back-door criterion for DAGs, MAGs, CPDAGs and PAGs, where CPDAGs and PAGs represent Markov equivalence classes of DAGs or MAGs, respectively, and can be inferred directly from data (see, e.g., Spirtes et al., 2000; Chickering, 2003; Colombo et al., 2012; Claassen et al., 2013; Colombo and Maathuis, 2014). The generalized back-door criterion is sufficient but not necessary for adjustment.

In this paper, we extend the results of Shpitser et al. (2012), van der Zander et al. (2014) and Maathuis and Colombo (2015) to derive a single necessary and sufficient adjustment criterion that holds for all four graph classes: DAGs, CPDAGs, MAGs and PAGs.

To illustrate the use of our generalized adjustment criterion, suppose we are given the CPDAG in Figure 1a and we want to estimate the total causal effect of  $X$  on  $Y$ . Our criterion will inform us that the set  $\{A, Z\}$  is an adjustment set for this CPDAG, which means that it is an adjustment set in every DAG that the CPDAG represents (Figure 1b). Hence, we can estimate the causal effect without knowledge of the full causal structure. In a similar manner, by applying our criterion to a MAG or a PAG, we find adjustment sets that are valid for all DAGs represented by this MAG or PAG. Our criterion finds such adjustment sets whenever they exist; else, our knowledge of the model structure is insufficient to compute the desired causal effect by covariate adjustment. We hope that this ability to allow for incomplete structural knowledge or latent confounding or both will help address concerns that graphical causal modelling “assumes that all [...] DAGs have been properly specified” (West and Koch, 2014).

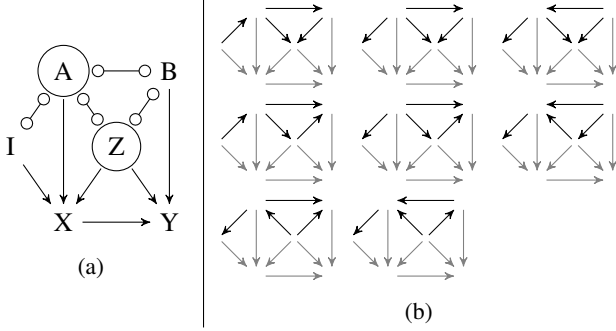


Figure 1: (a) A CPDAG in which, according to our criterion,  $\{A, Z\}$  is an adjustment set for the total causal effect of  $X$  on  $Y$ . (b) The Markov equivalence class of (a), with node labels removed for simplicity and varying edges highlighted. An adjustment set for a CPDAG (PAG) is one that works in all DAGs (MAGs) of the Markov equivalence class.

We note that, although we can find all causal effects that are identifiable by covariate adjustment, we generally do not find all identifiable causal effects, since some effects may be identifiable by other means, such as Pearl’s front-door criterion (Pearl, 2009, Section 3.3.2) or the ID algorithm (Tian and Pearl, 2002; Shpitser and Pearl, 2006). We also point out that MAGs and PAGs are in principle not only able to represent unobserved confounding, but can also account for unobserved selection variables. However, in this paper we assume that there are no unobserved selection variables. This restriction is mainly due to the fact that selection bias often renders it impossible to identify causal effects using just covariate adjustment. Bareinboim et al. (2014) discuss these problems and present creative approaches to work around them, e.g., by combining data from different sources. We leave the question whether adjustment could be combined with such auxiliary methods aside for future research.

## 2 PRELIMINARIES

Throughout the paper we denote sets in bold uppercase letters (e.g.,  $\mathbf{S}$ ), graphs in calligraphic font (e.g.,  $\mathcal{G}$ ) and nodes in a graph in uppercase letters (e.g.,  $X$ )

**Nodes and edges.** A graph  $\mathcal{G} = (\mathbf{V}, \mathbf{E})$  consists of a set of nodes (variables)  $\mathbf{V} = \{X_1, \dots, X_p\}$  and a set of edges  $\mathbf{E}$ .

There is at most one edge between any pair of nodes, and nodes are called *adjacent* if they are connected by an edge. Every edge has two edge marks that can be arrowheads, tails or circles. Edges can be *directed*  $\rightarrow$ , *bidirected*  $\leftrightarrow$ , *non-directed*  $\circ\text{--}\circ$  or *partially directed*  $\circ\text{--}\rightarrow$ . We use  $\bullet$  as a stand in for any of the allowed edge marks. An edge is *into* (*out of*) a node  $X$  if the edge has an arrowhead (tail) at  $X$ . A *directed graph* contains only directed edges. A

*mixed graph* may contain directed and bi-directed edges. A *partial mixed graph* may contain any of the described edges. Unless stated otherwise, all definitions apply for partial mixed graphs.

**Paths.** A *path*  $p$  from  $X$  to  $Y$  in  $\mathcal{G}$  is a sequence of distinct nodes  $\langle X, \dots, Y \rangle$  in which every pair of successive nodes is adjacent in  $\mathcal{G}$ . A node  $V$  *lies on a path*  $p$  if  $V$  occurs in the sequence of nodes. The *length* of a path equals the number of edges on the path. A *directed path* from  $X$  to  $Y$  is a path from  $X$  to  $Y$  in which all edges are directed towards  $Y$ , i.e.,  $X \rightarrow \dots \rightarrow Y$ . A directed path is also called a *causal path*. A *possibly directed path* (*possibly causal path*) from  $X$  to  $Y$  is a path from  $X$  to  $Y$  that has no arrowhead pointing to  $X$ . A path from  $X$  to  $Y$  that is not possibly causal is called a *non-causal path* from  $X$  to  $Y$ . A directed path from  $X$  to  $Y$  together with an edge  $Y \rightarrow X$  ( $Y \leftrightarrow X$ ) forms an (*almost*) *directed cycle*. For two disjoint subsets  $\mathbf{X}$  and  $\mathbf{Y}$  of  $\mathbf{V}$ , a path from  $\mathbf{X}$  to  $\mathbf{Y}$  is a path from some  $X \in \mathbf{X}$  to some  $Y \in \mathbf{Y}$ . A path from  $\mathbf{X}$  to  $\mathbf{Y}$  is *proper* if only its first node is in  $\mathbf{X}$ .

**Subsequences and subpaths.** A *subsequence* of a path  $p$  is a sequence of nodes obtained by deleting some nodes from  $p$  without changing the order of the remaining nodes. A subsequence of a path is not necessarily a path. For a path  $p = \langle X_1, X_2, \dots, X_m \rangle$ , the *subpath* from  $X_i$  to  $X_k$  ( $1 \leq i \leq k \leq m$ ) is the path  $p(X_i, X_k) = \langle X_i, X_{i+1}, \dots, X_k \rangle$ . We denote the concatenation of paths by  $\oplus$ , so that for example  $p = p(X_1, X_k) \oplus p(X_k, X_m)$ . We use the convention that we remove any loops that may occur due to the concatenation, so that the result is again a path.

**Ancestral relationships.** If  $X \rightarrow Y$ , then  $X$  is a *parent* of  $Y$ . If there is a (possibly) directed path from  $X$  to  $Y$ , then  $X$  is a (*possible*) *ancestor* of  $Y$ , and  $Y$  is a (*possible*) *descendant* of  $X$ . Every node is also a descendant and an ancestor of itself. The sets of parents and (possible) descendants of  $X$  in  $\mathcal{G}$  are denoted by  $\text{Pa}(X, \mathcal{G})$  and  $(\text{Poss})\text{De}(X, \mathcal{G})$  respectively. For a set of nodes  $\mathbf{X} \subseteq \mathbf{V}$ , we have  $\text{Pa}(\mathbf{X}, \mathcal{G}) = \cup_{X \in \mathbf{X}} \text{Pa}(X, \mathcal{G})$ , with analogous definitions for  $(\text{Poss})\text{De}(\mathbf{X}, \mathcal{G})$ .

**Colliders and shields.** If a path  $p$  contains  $X_i \bullet \rightarrow X_j \leftarrow \bullet X_k$  as a subpath, then  $X_j$  is a *collider* on  $p$ . A *collider path* is a path on which every non-endpoint node is a collider. A path of length one is a trivial collider path. A path  $\langle X_i, X_j, X_k \rangle$  is an (*un*)*shielded triple* if  $X_i$  and  $X_k$  are (not) adjacent. A path is *unshielded* if all successive triples on the path are unshielded. Otherwise the path is *shielded*. A node  $X_j$  is a *definite non-collider* on a path  $p$  if there is at least one edge out of  $X_j$  on  $p$ , or if  $X_i \bullet \circ X_j \circ \bullet X_k$  is a subpath of  $p$  and  $\langle X_i, X_j, X_k \rangle$  is an unshielded triple. A node is of *definite status* on a path if it is a collider or a definite non-collider on the path. A path  $p$  is of definite status if every non-endpoint node on  $p$  is of definite status. An unshielded path is always of definite

status, but a definite status path is not always unshielded.

**m-separation and m-connection.** A definite status path  $p$  between nodes  $X$  and  $Y$  is *m-connecting* given a set of nodes  $\mathbf{Z}$  ( $X, Y \notin \mathbf{Z}$ ) if every definite non-collider on  $p$  is not in  $\mathbf{Z}$ , and every collider on  $p$  has a descendant in  $\mathbf{Z}$ . Otherwise  $\mathbf{Z}$  blocks  $p$ . If  $\mathbf{Z}$  blocks all definite status paths between  $X$  and  $Y$ , we say that  $X$  and  $Y$  are m-separated given  $\mathbf{Z}$ . Otherwise,  $X$  and  $Y$  are m-connected given  $\mathbf{Z}$ . For pairwise disjoint subsets  $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$  of  $\mathbf{V}$ ,  $\mathbf{X}$  and  $\mathbf{Y}$  are m-separated given  $\mathbf{Z}$  if  $X$  and  $Y$  are m-separated by  $\mathbf{Z}$  for any  $X \in \mathbf{X}$  and  $Y \in \mathbf{Y}$ . Otherwise,  $\mathbf{X}$  and  $\mathbf{Y}$  are m-connected given  $\mathbf{Z}$ .

**Causal Bayesian networks.** A directed graph without directed cycles is a *directed acyclic graph* (DAG). A Bayesian network for a set of variables  $\mathbf{V} = \{X_1, \dots, X_p\}$  is a pair  $(\mathcal{G}, f)$ , where  $\mathcal{G}$  is a DAG, and  $f$  is a joint probability density for  $\mathbf{V}$  that factorizes according to the conditional independence relationships described via m-separation, that is  $f(\mathbf{V}) = \prod_{i=1}^p f(X_i | Pa(X_i, \mathcal{G}))$  (Pearl, 2009). We call a DAG causal when every edge  $X_i \rightarrow X_j$  in  $\mathcal{G}$  represents a direct causal effect of  $X_i$  on  $X_j$ . A Bayesian network  $(\mathcal{G}, f)$  is a *causal Bayesian network* if  $\mathcal{G}$  is a causal DAG. If a causal Bayesian network is given and all variables are observed one can easily derive post-intervention densities. In particular, we consider interventions  $do(\mathbf{X} = \mathbf{x})$  ( $\mathbf{X} \subseteq \mathbf{V}$ ), which represent outside interventions that set  $\mathbf{X}$  to  $\mathbf{x}$  (Pearl, 2009):

$$f(\mathbf{v} | do(\mathbf{X} = \mathbf{x})) = \begin{cases} \prod_{X_i \in \mathbf{V} \setminus \mathbf{X}} f(x_i | Pa(x_i, \mathcal{G})), & \text{for values of } \mathbf{V} \\ & \text{consistent with } \mathbf{x}, \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

Equation (1) is known as the truncated factorization formula (Pearl, 2009) or the g-formula (Robins, 1986).

**Maximal ancestral graph.** A mixed graph  $\mathcal{G}$  without directed cycles and almost directed cycles is called *ancestral*. A *maximal ancestral graph* (MAG) is an ancestral graph  $\mathcal{G} = (\mathbf{V}, \mathbf{E})$  where every two non-adjacent nodes  $X$  and  $Y$  in  $\mathcal{G}$  can be m-separated by a set  $\mathbf{Z} \subseteq \mathbf{V} \setminus \{X, Y\}$ . A DAG with unobserved variables can be uniquely represented by a MAG that preserves the ancestral and m-separation relationships among the observed variables (Richardson and Spirtes, 2002). The MAG of a causal DAG is a *causal MAG*.

**Markov equivalence.** Several DAGs can encode the same conditional independence information via m-separation. Such DAGs form a *Markov equivalence class* which can be described uniquely by a *completed partially directed acyclic graph* (CPDAG). Several MAGs can also encode the same conditional independence information. Such MAGs form a Markov equivalence class which can be

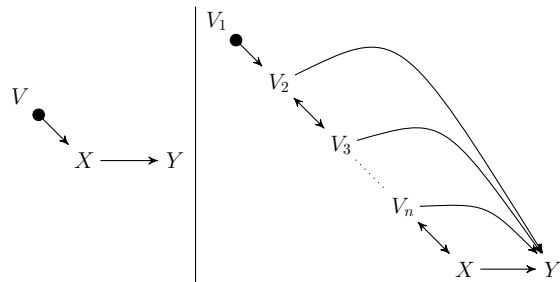


Figure 2: Two configurations where the edge  $X \rightarrow Y$  is visible.

described uniquely by a *partial ancestral graph* (PAG) (Richardson and Spirtes, 2002; Ali et al., 2009). We denote all DAGs (MAGs) in the Markov equivalence class described by a CPDAG (PAG)  $\mathcal{G}$  by  $[\mathcal{G}]$ .

**Consistent density.** A density  $f$  is *consistent* with a causal DAG  $\mathcal{D}$  if the pair  $(\mathcal{D}, f)$  forms a causal Bayesian network. A density  $f$  is consistent with a causal MAG  $\mathcal{M}$  if there exists a causal Bayesian network  $(\mathcal{D}', f')$ , such that  $\mathcal{M}$  represents  $\mathcal{D}'$  and  $f$  is the observed marginal of  $f'$ . A density  $f$  is consistent with a CPDAG (PAG)  $\mathcal{G}$  if it is consistent with a DAG (MAG) in  $[\mathcal{G}]$ .

**Visible and invisible edges.** All directed edges in DAGs and CPDAGs are said to be visible. Given a MAG  $\mathcal{M}$  or a PAG  $\mathcal{G}$ , a directed edge  $X \rightarrow Y$  is *visible* if there is a node  $V$  not adjacent to  $Y$  such that there is an edge between  $V$  and  $X$  that is into  $X$ , or if there is a collider path from  $V$  to  $X$  that is into  $X$  and every non-endpoint node on the path is a parent of  $Y$ . Otherwise,  $X \rightarrow Y$  is said to be *invisible* (Zhang, 2006; Maathuis and Colombo, 2015).

A directed visible edge  $X \rightarrow Y$  means that there are no latent confounders between  $X$  and  $Y$ .

### 3 MAIN RESULT

Throughout, let  $\mathcal{G} = (\mathbf{V}, \mathbf{E})$  represent a DAG, CPDAG, MAG or PAG, and let  $\mathbf{X}, \mathbf{Y}$  and  $\mathbf{Z}$  be pairwise disjoint subsets of  $\mathbf{V}$ , with  $\mathbf{X} \neq \emptyset$  and  $\mathbf{Y} \neq \emptyset$ . Here  $\mathbf{X}$  represents the intervention variables and  $\mathbf{Y}$  represents the set of response variables, i.e., we are interested in the causal effect of  $\mathbf{X}$  on  $\mathbf{Y}$ .

We define sound and complete graphical conditions for adjustment sets relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{G}$ . Thus, if a set  $\mathbf{Z}$  satisfies our conditions relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{G}$  (Definition 3.3), then it is a valid adjustment set for calculating the causal effect of  $\mathbf{X}$  on  $\mathbf{Y}$  (Definition 3.1), and every existing valid adjustment set satisfies our conditions (see Theorem 3.4). First, we define what we mean by an adjustment set.

**Definition 3.1. (Adjustment set; Maathuis and Colombo, 2015)** Let  $\mathcal{G}$  represent a DAG, CPDAG, MAG or PAG. Then  $\mathbf{Z}$  is an adjustment set relative to  $(\mathbf{X}, \mathbf{Y})$

in  $\mathcal{G}$  if for any density  $f$  consistent with  $\mathcal{G}$  we have  $f(\mathbf{y}|do(\mathbf{x})) = \begin{cases} f(\mathbf{y}|\mathbf{x}) & \text{if } \mathbf{Z} = \emptyset, \\ \int_{\mathbf{Z}} f(\mathbf{y}|\mathbf{x}, \mathbf{z})f(\mathbf{z})d\mathbf{z} = E_{\mathbf{Z}}\{f(\mathbf{y}|\mathbf{z}, \mathbf{x})\} & \text{otherwise.} \end{cases}$  If  $\mathbf{X} = \{X\}$  and  $\mathbf{Y} = \{Y\}$ , we call  $\mathbf{Z}$  an adjustment set relative to  $(X, Y)$  in the given graph.

To define our generalized adjustment criterion, we introduce the concept of amenability:

**Definition 3.2. (Amenability for DAGs, CPDAGs, MAGs and PAGs)** A DAG, CPDAG, MAG or PAG  $\mathcal{G}$  is said to be adjustment amenable, relative to  $(\mathbf{X}, \mathbf{Y})$  if every possibly directed proper path from  $\mathbf{X}$  to  $\mathbf{Y}$  in  $\mathcal{G}$  starts with a visible edge out of  $\mathbf{X}$ .

For conciseness, we will also write “amenable” instead of “adjustment amenable”. The intuition behind the concept of amenability is the following. In MAGs and PAGs, directed edges  $X \rightarrow Y$  can represent causal effects, but also mixtures of causal effects and latent confounding; in CPDAGs and PAGs, there are edges with unknown direction. This complicates adjustment because paths containing such edges can correspond to causal paths in some represented DAGs and to non-causal paths in others. For instance, when the graph  $X \rightarrow Y$  is interpreted as a DAG, the empty set is a valid adjustment set with respect to  $(X, Y)$  because there is only one path from  $X$  to  $Y$ , which is causal. When the same graph is however interpreted as a MAG, it can still represent the DAG  $X \rightarrow Y$ , but also for example the DAG  $X \leftarrow L \rightarrow Y$  where  $L$  is latent. A similar problem arises in the CPDAG  $X \circ - \circ Y$ .

We will show that for a graph  $\mathcal{G}$  that is not amenable relative to  $(\mathbf{X}, \mathbf{Y})$ , there is no adjustment set relative to  $(\mathbf{X}, \mathbf{Y})$  in the sense of Definition 3.1 (see Lemma 5.2). Note that every DAG is amenable, since all edges in a DAG are visible and directed. For MAGs, our notion of amenability reduces to the one defined by van der Zander et al. (2014).

We now introduce our Generalized Adjustment Criterion (GAC) for DAGs, CPDAGs, MAGs and PAGs.

**Definition 3.3. (Generalized Adjustment Criterion (GAC))** Let  $\mathcal{G}$  represent a DAG, CPDAG, MAG or PAG. Then  $\mathbf{Z}$  satisfies the generalized adjustment criterion relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{G}$  if the following three conditions hold:

- (0)  $\mathcal{G}$  is adjustment amenable relative to  $(\mathbf{X}, \mathbf{Y})$ , and
- (1) no element in  $\mathbf{Z}$  is a possible descendant in  $\mathcal{G}$  of any  $W \in \mathbf{V} \setminus \mathbf{X}$  which lies on a proper possibly causal path from  $\mathbf{X}$  to  $\mathbf{Y}$ , and
- (2) all proper definite status non-causal paths in  $\mathcal{G}$  from  $\mathbf{X}$  to  $\mathbf{Y}$  are blocked by  $\mathbf{Z}$ .

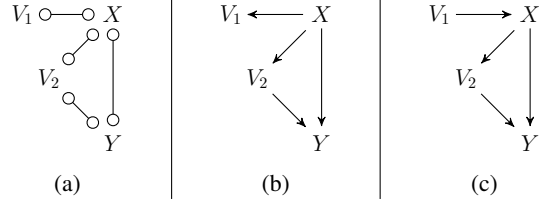


Figure 3: (a) PAG  $\mathcal{P}$ , (b) MAG  $\mathcal{M}_1$ , (c) MAG  $\mathcal{M}_2$  used in Example 4.2.

Note that condition (0) does not depend on  $\mathbf{Z}$ . In other words, if condition (0) is violated, then there is no set  $\mathbf{Z}' \subseteq \mathbf{V} \setminus (\mathbf{X} \cup \mathbf{Y})$  that satisfies the generalized adjustment criterion relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{G}$ .

Condition (1) defines a set of nodes that cannot be used in an adjustment set. Denoting this set of forbidden nodes by

$$\mathbf{F}_{\mathcal{G}}(\mathbf{X}, \mathbf{Y}) = \{W' \in \mathbf{V} : W' \in \text{PossDe}(W, \mathcal{G}) \text{ for some } W \notin \mathbf{X} \text{ which lies on a proper possibly causal path from } \mathbf{X} \text{ to } \mathbf{Y}\}, \quad (2)$$

condition (1) can be stated as:  $\mathbf{Z} \cap \mathbf{F}_{\mathcal{G}}(\mathbf{X}, \mathbf{Y}) = \emptyset$ . We will sometimes use this notation in examples and proofs.

We now give the main theorem of this paper.

**Theorem 3.4.** Let  $\mathcal{G}$  represent a DAG, CPDAG, MAG or PAG. Then  $\mathbf{Z}$  is an adjustment set relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{G}$  (Definition 3.1) if and only if  $\mathbf{Z}$  satisfies the generalized adjustment criterion relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{G}$  (Definition 3.3).

## 4 EXAMPLES

We now provide some examples that illustrate how the generalized adjustment criterion can be applied.

**Example 4.1.** We first return to the example of the Introduction. Consider the CPDAG  $\mathcal{C}$  in Figure 1a. Note that  $\mathcal{C}$  is amenable relative to  $(X, Y)$  and that  $\mathbf{F}_{\mathcal{C}}(X, Y) = \{Y\}$ . Hence, any node other than  $X$  and  $Y$  can be used in an adjustment set. Note that every definite status non-causal path  $p$  from  $X$  to  $Y$  has one of the following paths as a subsequence:  $p_1 = \langle X, Z, Y \rangle$  and  $p_2 = \langle X, A, B, Y \rangle$ , and nodes on  $p$  that are not on  $p_1$  or  $p_2$  are non-colliders on  $p$ . Hence, if we block  $p_1$  and  $p_2$ , then we block all definite status non-causal paths from  $X$  to  $Y$ . This implies that any superset of  $\{Z, A\}$  and  $\{Z, B\}$  is an adjustment set relative to  $(X, Y)$  in  $\mathcal{C}$ , and all adjustment sets are given by:  $\{Z, A\}$ ,  $\{Z, B\}$ ,  $\{Z, A, I\}$ ,  $\{Z, B, I\}$ ,  $\{Z, A, B\}$  and  $\{Z, A, B, I\}$ .

**Example 4.2.** To illustrate the concept of amenability, consider Figure 3 with a PAG  $\mathcal{P}$  in (a), and two MAGs  $\mathcal{M}_1$  and  $\mathcal{M}_2$  in  $[\mathcal{P}]$  in (b) and (c). Note that  $\mathcal{P}$  and  $\mathcal{M}_1$  are not amenable relative to  $(X, Y)$ . For  $\mathcal{P}$  this is due to the

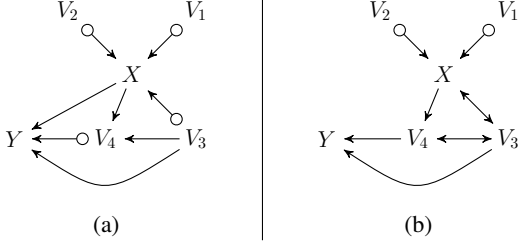


Figure 4: (a) PAG  $\mathcal{P}_1$ , (b) PAG  $\mathcal{P}_2$  used in Example 4.3.

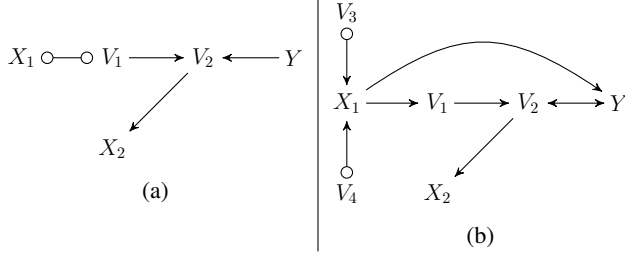


Figure 5: (a) CPDAG  $\mathcal{C}$ , (b) PAG  $\mathcal{P}$  used in Example 4.4.

path  $X \circ - \circ Y$ , and for  $\mathcal{M}_1$  this is due to the invisible edge  $X \rightarrow Y$ . On the other hand,  $\mathcal{M}_2$  is amenable relative to  $(X, Y)$ , since the edges  $X \rightarrow Y$  and  $X \rightarrow V_2$  are visible due to the edge  $V_1 \rightarrow X$  and the fact that  $V_1$  is not adjacent to  $Y$  or  $V_2$ . Since there are no proper definite status non-causal paths from  $X$  to  $Y$  in  $\mathcal{M}_2$ , it follows that the empty set satisfies the generalized adjustment criterion relative to  $(X, Y)$  in  $\mathcal{M}_2$ . Finally, note that  $\mathcal{M}_1$  could also be interpreted as a DAG. In that case it would be amenable relative to  $(X, Y)$ . This shows that amenability depends crucially on the interpretation of the graph.

**Example 4.3.** Let  $\mathcal{P}_1$  and  $\mathcal{P}_2$  be the PAGs in Figure 4(a) and Figure 4(b), respectively. Both PAGs are amenable relative to  $(X, Y)$ . We will show that there is an adjustment set relative to  $(X, Y)$  in  $\mathcal{P}_1$  but not in  $\mathcal{P}_2$ . This illustrates that amenability is not a sufficient criterion for the existence of an adjustment set.

We first consider  $\mathcal{P}_1$ . Note that  $\mathbf{F}_{\mathcal{P}_1}(X, Y) = \{V_4, Y\}$  is the set of nodes that cannot be used for adjustment. There are two proper definite status non-causal paths from  $X$  to  $Y$  in  $\mathcal{P}_1$ :  $X \leftarrow \circ V_3 \rightarrow Y$  and  $X \rightarrow V_4 \leftarrow V_3 \rightarrow Y$ . These are blocked by any set containing  $V_3$ . Hence, all sets satisfying the GAC relative to  $(X, Y)$  in  $\mathcal{P}_1$  are:  $\{V_3\}$ ,  $\{V_1, V_3\}$ ,  $\{V_2, V_3\}$  and  $\{V_1, V_2, V_3\}$ .

We now consider  $\mathcal{P}_2$ . Note that  $\mathbf{F}_{\mathcal{P}_2}(X, Y) = \mathbf{F}_{\mathcal{P}_1}(X, Y) = \{V_4, Y\}$ . There are three proper definite status non-causal paths from  $X$  to  $Y$  in  $\mathcal{P}_2$ :  $p_1 = X \leftrightarrow V_3 \rightarrow Y$ ,  $p_2 = X \leftrightarrow V_3 \leftrightarrow V_4 \rightarrow Y$  and  $p_3 = X \rightarrow V_4 \leftrightarrow V_3 \rightarrow Y$ . To block  $p_1$ , we must also use  $V_3$ . This implies that we must use  $V_4$  to block  $p_2$ . But  $V_4 \in \mathbf{F}_{\mathcal{P}_2}(X, Y)$ . Hence, there is no set  $\mathbf{Z}$  that satisfies the GAC relative to  $(X, Y)$  in  $\mathcal{P}_2$ .

**Example 4.4.** Let  $\mathbf{X} = \{X_1, X_2\}$  and  $\mathbf{Y} = \{Y\}$  and consider the CPDAG  $\mathcal{C}$  and the PAG  $\mathcal{P}$  in Figures 5(a) and 5(b). We will show that for both graphs there is no set that satisfies the generalized back-door criterion of Maathuis and Colombo (2015) relative to  $(\mathbf{X}, \mathbf{Y})$ , but there are sets that satisfy the generalized adjustment criterion relative to  $(\mathbf{X}, \mathbf{Y})$  in these graphs.

Recall that a set  $\mathbf{Z}$  satisfies the generalized back-door criterion relative to  $(\mathbf{X}, \mathbf{Y})$  and a CPDAG (PAG)  $\mathcal{G}$  if  $\mathbf{Z}$  contains no possible descendants of  $\mathbf{X}$  in  $\mathcal{G}$  and if for every  $X \in \mathbf{X}$  the set  $\mathbf{Z} \cup \mathbf{X} \setminus \{X\}$  blocks every definite status path from  $X$  to every  $Y \in \mathbf{Y}$  in  $\mathcal{G}$  that does not start with a visible edge out of  $X$ .

We first consider the CPDAG  $\mathcal{C}$ . To block the path  $X_2 \leftarrow V_2 \leftarrow Y$ , we must use node  $V_2$ , but  $V_2 \in \text{PossDe}(X_1, \mathcal{C})$ . Hence, no set  $\mathbf{Z}$  can satisfy the generalized back-door criterion relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{C}$ . However,  $\{V_1, V_2\}$  satisfies the generalized adjustment criterion relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{C}$ .

We now consider  $\mathcal{P}$ . To block the path  $X_2 \leftarrow V_2 \leftrightarrow Y$ , we must use node  $V_2$ . But,  $V_2 \in \text{De}(X_1, \mathcal{P})$  and thus there is no set satisfying the generalized back-door criterion relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{P}$ . However, sets  $\{V_1, V_2\}$ ,  $\{V_1, V_2, V_3\}$ ,  $\{V_1, V_2, V_4\}$ ,  $\{V_1, V_2, V_3, V_4\}$  all satisfy the generalized adjustment criterion relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{P}$ .

## 5 PROOF OF THEOREM 3.4

For DAGs and MAGs, our generalized adjustment criterion reduces to the following adjustment criterion:

**Definition 5.1. (Adjustment Criterion (AC))** Let  $\mathcal{G} = (\mathbf{V}, \mathbf{E})$  represent a DAG or MAG. Then  $\mathbf{Z}$  satisfies the adjustment criterion relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{G}$  if the following three conditions hold:

- (0\*)  $\mathcal{G}$  is adjustment amenable with respect to  $(\mathbf{X}, \mathbf{Y})$ , and
  - (a) no element in  $\mathbf{Z}$  is a descendant in  $\mathcal{G}$  of any  $W \in \mathbf{V} \setminus \mathbf{X}$  which lies on a proper causal path from  $\mathbf{X}$  to  $\mathbf{Y}$ , and
  - (b) all proper non-causal paths in  $\mathcal{G}$  from  $\mathbf{X}$  to  $\mathbf{Y}$  are blocked by  $\mathbf{Z}$ .

This adjustment criterion is a slightly reformulated but equivalent version of the adjustment criterion of Shpitser et al. (2012) for DAGs and of van der Zander et al. (2014) for MAGs, with amenability directly included in the criterion. This adjustment criterion was shown to be sound and complete for DAGs (Shpitser et al., 2012; Shpitser, 2012) and MAGs (van der Zander et al., 2014). We therefore only need to prove Theorem 3.4 for CPDAGs and PAGs.

To this end, we need three main lemmas, given below. Throughout, we let  $\mathcal{G} = (\mathbf{V}, \mathbf{E})$  represent a CPDAG or



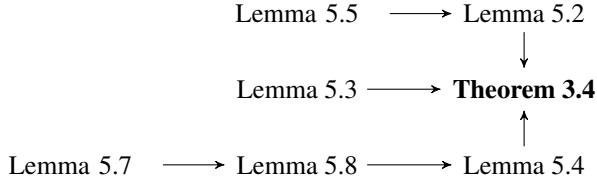


Figure 6: Proof structure of Theorem 3.4.

a PAG, and we let  $\mathbf{X}$ ,  $\mathbf{Y}$  and  $\mathbf{Z}$  be pairwise disjoint subsets of  $\mathbf{V}$ , with  $\mathbf{X} \neq \emptyset$  and  $\mathbf{Y} \neq \emptyset$ . We use GAC and AC to refer to the generalized adjustment criterion (Definition 3.3) and adjustment criterion (Definition 5.1), respectively.

Lemma 5.2 is about condition (0) of the GAC:

**Lemma 5.2.** *If a CPDAG (PAG)  $\mathcal{G}$  satisfies condition (0) of the GAC relative to  $(\mathbf{X}, \mathbf{Y})$ , then every DAG (MAG) in  $[\mathcal{G}]$  satisfies condition (0\*) of the AC relative to  $(\mathbf{X}, \mathbf{Y})$ . On the other hand, if  $\mathcal{G}$  violates condition (0) of the GAC relative to  $(\mathbf{X}, \mathbf{Y})$ , then there exists no set  $\mathbf{Z}' \subseteq \mathbf{V} \setminus (\mathbf{X} \cup \mathbf{Y})$  that is an adjustment set relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{G}$  (see Definition 3.1).*

Next, we assume that  $\mathcal{G}$  satisfies condition (0) of the GAC relative to  $(\mathbf{X}, \mathbf{Y})$ . Under this assumption, we show that  $\mathbf{Z}$  satisfies conditions (1) and (2) of the GAC relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{G}$  if and only if  $\mathbf{Z}$  satisfies conditions (a) and (b) of the AC relative to  $(\mathbf{X}, \mathbf{Y})$  in every DAG (MAG) in  $[\mathcal{G}]$ . This is shown in two separate lemmas:

**Lemma 5.3.** *Let condition (0) of the GAC be satisfied relative to  $(\mathbf{X}, \mathbf{Y})$  in a CPDAG (PAG)  $\mathcal{G}$ . Then the following two statements are equivalent:*

- $\mathbf{Z}$  satisfies condition (1) of the GAC relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{G}$ .
- $\mathbf{Z}$  satisfies condition (a) of the AC relative to  $(\mathbf{X}, \mathbf{Y})$  in every DAG (MAG) in  $[\mathcal{G}]$ .

**Lemma 5.4.** *Let condition (0) of the GAC be satisfied relative to  $(\mathbf{X}, \mathbf{Y})$  in a CPDAG (PAG)  $\mathcal{G}$ , and let  $\mathbf{Z}$  satisfy condition (1) of the GAC relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{G}$ . Then the following two statements are equivalent:*

- $\mathbf{Z}$  satisfies condition (2) of the GAC relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{G}$ .
- $\mathbf{Z}$  satisfies condition (b) of the AC relative to  $(\mathbf{X}, \mathbf{Y})$  in every DAG (MAG) in  $[\mathcal{G}]$ .

The proofs of Lemmas 5.2, 5.3 and 5.4 are discussed in Sections 5.1, 5.2 and 5.3, respectively. Some proofs require additional lemmas that can be found in the supplement. The proof of Lemma 5.4 is the most technical, and builds on the work of Zhang (2006).

Figure 6 shows how all lemmas fit together to prove Theorem 3.4.

**Proof of Theorem 3.4:** First, suppose that the CPDAG (PAG)  $\mathcal{G}$  and the sets  $\mathbf{X}$ ,  $\mathbf{Y}$  and  $\mathbf{Z}$  satisfy all conditions of the GAC. By applying Lemmas 5.2, 5.3 and 5.4 in turn, it directly follows that all conditions of the AC are satisfied by  $\mathbf{X}$ ,  $\mathbf{Y}$  and  $\mathbf{Z}$  and any DAG (MAG) in  $[\mathcal{G}]$ .

To prove the other direction, suppose that the tuple  $\mathcal{G}$ ,  $\mathbf{X}$ ,  $\mathbf{Y}$ ,  $\mathbf{Z}$  does not satisfy all conditions of the GAC. First, suppose that  $\mathcal{G}$  violates condition (0) relative to  $(\mathbf{X}, \mathbf{Y})$ . Then by Lemma 5.2, there is no adjustment set relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{G}$ , and hence  $\mathbf{Z}$  is certainly not an adjustment set.

Otherwise,  $\mathbf{Z}$  must violate condition (1) or (2) of the GAC relative to  $(\mathbf{X}, \mathbf{Y})$ . By applying Lemmas 5.3 and 5.4 in turn, this implies that there is a DAG  $\mathcal{D}$  (MAG  $\mathcal{M}$ ) in  $[\mathcal{G}]$  such that  $\mathbf{Z}$  violates conditions (a) or (b) of the AC relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{D}$  ( $\mathcal{M}$ ). Since the AC is sound and complete for DAGs and MAGs, this implies that  $\mathbf{Z}$  is not an adjustment set relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{D}$  ( $\mathcal{M}$ ), so that  $\mathbf{Z}$  is certainly not an adjustment set relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{G}$ . □

## 5.1 PROOF OF LEMMA 5.2

The proof of Lemma 5.2 is based on the following lemma:

**Lemma 5.5.** *Let  $X$  and  $Y$  be nodes in a PAG  $\mathcal{P}$ , such that there is a possibly directed path  $p^*$  from  $X$  to  $Y$  in  $\mathcal{P}$  that does not start with a visible edge out of  $X$ . Then there is a MAG  $\mathcal{M}$  in  $[\mathcal{P}]$  such that the path  $p$  in  $\mathcal{M}$ , consisting of the same sequence of nodes as  $p^*$  in  $\mathcal{P}$ , contains a subsequence that is a directed path from  $X$  to  $Y$  starting with an invisible edge in  $\mathcal{M}$ .*

The proof of Lemma 5.5 is given in the supplement.

**Proof of Lemma 5.2:** First suppose that  $\mathcal{G}$  satisfies condition (0) of the GAC relative to  $(\mathbf{X}, \mathbf{Y})$ , meaning that every proper possibly directed path from  $\mathbf{X}$  to  $\mathbf{Y}$  in  $\mathcal{G}$  starts with a visible edge out of  $\mathbf{X}$ . Any visible edge in  $\mathcal{G}$  is visible in all DAGs (MAGs) in  $[\mathcal{G}]$ , and any proper directed path in a DAG (MAG) in  $[\mathcal{G}]$  corresponds to a proper possibly directed path in  $\mathcal{G}$ . Hence, any proper directed path from  $\mathbf{X}$  to  $\mathbf{Y}$  in any DAG (MAG) in  $[\mathcal{G}]$  starts with a visible edge out of  $\mathbf{X}$ . This shows that all DAGs (MAGs) in  $[\mathcal{G}]$  satisfy condition (0\*) of the AC relative to  $(\mathbf{X}, \mathbf{Y})$ .

Next, suppose that  $\mathcal{G}$  violates condition (0) of the GAC relative to  $(\mathbf{X}, \mathbf{Y})$ . We will show that this implies that there is no set  $\mathbf{Z}' \subseteq \mathbf{V} \setminus (\mathbf{X} \cup \mathbf{Y})$  that is an adjustment set relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{G}$ . We give separate proofs for CPDAGs and PAGs.

Thus, let  $\mathcal{G}$  represent a CPDAG and suppose that there is a proper possibly directed path  $p$  from a node  $X \in \mathbf{X}$  to a node  $Y \in \mathbf{Y}$  that starts with a non-directed edge ( $\circ \rightarrow \circ$ ).

Let  $p' = \langle X, V_1, \dots, Y \rangle$  (where  $V_1 = Y$  is allowed) be a shortest subsequence of  $p$  such that  $p'$  is also a proper possibly directed path from  $X$  to  $Y$  starting with a non-directed edge in  $\mathcal{G}$ . We first show that  $p'$  is a definite status path, by contradiction. Thus, suppose that  $p'$  is not a definite status path. Then the length of  $p'$  is at least 2, and we write  $p' = \langle X, V_1, \dots, V_k = Y \rangle$  for  $k \geq 2$ . Since the subpath  $p'(V_1, Y)$  is a definite status path (otherwise we can choose a shorter path), this means that  $V_1$  is not of a definite status on  $p'$ . This implies the existence of an edge between  $X$  and  $V_2$ . This edge must be of the form  $X \rightarrow V_2$ , since  $X \circ\text{-} V_2$  implies that we can choose a shorter path, and  $X \leftarrow V_2$  together with  $X \circ\text{-} V_1$  implies  $V_1 \leftarrow V_2$  by Lemma 1 from Meek (1995) (see Section 1 of the supplement), so that  $p'$  is not possibly directed from  $X$  to  $Y$ . But the edge  $X \rightarrow V_2$  implies that  $V_1 \rightarrow V_2$ , since otherwise Lemma 1 from Meek (1995) implies  $X \rightarrow V_1$ . But then  $V_1$  is a definite non-collider on  $p'$ , which contradicts that  $V_1$  is not of definite status.

Hence,  $p'$  is a proper possibly directed definite status path from  $X$  to  $Y$ . By Lemma 7.6 from Maathuis and Colombo (2015) (see Section 1 of the supplement), there is a DAG  $\mathcal{D}_1$  in  $[\mathcal{G}]$  such that there are no additional arrowheads into  $X$ , as well as a DAG  $\mathcal{D}_2$  in  $[\mathcal{G}]$  such that there are no additional arrowheads into  $V_1$ . This means that the paths corresponding to  $p'$  are oriented as  $p'_1 = X \rightarrow V_1 \rightarrow \dots \rightarrow Y$  and  $p'_2 = X \leftarrow V_1 \rightarrow \dots \rightarrow Y$  in  $\mathcal{D}_1$  and  $\mathcal{D}_2$ . An adjustment set relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{D}_2$  must block the non-causal path  $p'_2$ , by using at least one of the non-endpoints nodes on this path. But all these nodes are in  $\mathbf{F}_{\mathcal{D}_1}(\mathbf{X}, \mathbf{Y})$  (see (2)). Hence, there is no set  $\mathbf{Z}' \subseteq \mathbf{V} \setminus (\mathbf{X} \cup \mathbf{Y})$  that satisfies the AC relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{D}_1$  and  $\mathcal{D}_2$  simultaneously. Since the AC is sound and complete for DAGs, this implies that there is no  $\mathbf{Z}' \subseteq \mathbf{V} \setminus (\mathbf{X} \cup \mathbf{Y})$  that is an adjustment set relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{G}$ .

Finally, let  $\mathcal{G}$  represent a PAG and suppose that there is a proper possibly directed path  $p$  from some  $X \in \mathbf{X}$  to some  $Y \in \mathbf{Y}$  that does not start with a visible edge out of  $X$  in  $\mathcal{G}$ .

By Lemma 5.5, there is a subsequence  $p'$  of  $p$  such that there is a MAG  $\mathcal{M}$  in  $[\mathcal{G}]$  where the corresponding path is directed from  $X$  to  $Y$  and starts with an invisible edge. Then  $\mathcal{M}$  is not amenable relative to  $(\mathbf{X}, \mathbf{Y})$ . By Lemma 5.7 from van der Zander et al. (2014) (see Section 1 of the supplement) this means that there is no set  $\mathbf{Z}' \subseteq \mathbf{V} \setminus (\mathbf{X} \cup \mathbf{Y})$  that is an adjustment set relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{M}$ . Hence, there is no set  $\mathbf{Z}' \subseteq \mathbf{V} \setminus (\mathbf{X} \cup \mathbf{Y})$  that is an adjustment set relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{G}$ .  $\square$

## 5.2 PROOF OF LEMMA 5.3

**Proof of Lemma 5.3:** First, suppose that  $\mathbf{Z}$  satisfies condition (1) of the GAC relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{G}$ . Then  $\mathbf{Z} \cap \mathbf{F}_{\mathcal{G}}(\mathbf{X}, \mathbf{Y}) = \emptyset$ . Since  $\mathbf{F}_{\mathcal{D}}(\mathbf{X}, \mathbf{Y}) \subseteq \mathbf{F}_{\mathcal{G}}(\mathbf{X}, \mathbf{Y})$

( $\mathbf{F}_{\mathcal{M}}(\mathbf{X}, \mathbf{Y}) \subseteq \mathbf{F}_{\mathcal{G}}(\mathbf{X}, \mathbf{Y})$ ) for any DAG  $\mathcal{D}$  (MAG  $\mathcal{M}$ ) in  $[\mathcal{G}]$ , it follows directly that  $\mathbf{Z}$  satisfies condition (a) of the AC relative to  $(\mathbf{X}, \mathbf{Y})$  in all DAGs (MAGs) in  $[\mathcal{G}]$ .

To prove the other direction, suppose that  $\mathcal{G}$  satisfies condition (0) of the GAC relative to  $(\mathbf{X}, \mathbf{Y})$ , but that  $\mathbf{Z}$  does not satisfy condition (1) of the GAC relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{G}$ . Then there is a node  $V \in \mathbf{Z} \cap \mathbf{F}_{\mathcal{G}}(\mathbf{X}, \mathbf{Y})$ , i.e.,  $V \in \mathbf{Z}$  and  $V$  is a possible descendant of a node  $W$  on a proper possibly directed path from some  $X \in \mathbf{X}$  to some  $Y \in \mathbf{Y}$  in  $\mathcal{G}$ . We denote this path by  $p = \langle X, V_1, \dots, V_k, Y \rangle$ , where  $k \geq 1$  and  $W \in \{V_1, \dots, V_k\}$ . Then the subpaths  $q = p(X, W)$  and  $r = p(W, Y)$  are also proper possibly directed paths. Moreover, there is a possibly directed path  $s$  from  $W$  to  $V$ , where this path is allowed to be of zero length (if  $W = V$ ). We will show that the existence of these paths implies that there is a DAG  $\mathcal{D}$  (MAG  $\mathcal{M}$ ) in  $[\mathcal{G}]$  such that  $\mathbf{Z}$  violates condition (a) of the AC relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{D}$  ( $\mathcal{M}$ ).

By Lemma B.1 from Zhang (2008) (see Section 1 of the supplement), there are subsequences  $q'$ ,  $r'$  and  $s'$  of  $q$ ,  $r$  and  $s$  that are unshielded proper possibly directed paths (again  $s'$  is allowed to be a path of zero length). Moreover,  $q'$  must start with a directed (visible) edge, since otherwise the concatenated path  $q' \oplus r'$ , which is again a proper possibly directed path from  $X$  to  $Y$ , would violate condition (0) of the GAC.

Lemma B.1 from Zhang (2008) then implies that  $q'$  is a directed path from  $X$  to  $W$  in  $\mathcal{G}$ . Hence, the path corresponding to  $q'$  is a directed path from  $X$  to  $W$  in any DAG (MAG) in  $[\mathcal{G}]$ .

By Lemma 7.6 from Maathuis and Colombo (2015), there is at least one DAG  $\mathcal{D}$  (MAG  $\mathcal{M}$ ) in  $[\mathcal{G}]$  that has no additional arrowheads into  $W$ . In this graph  $\mathcal{D}$  ( $\mathcal{M}$ ), the path corresponding to  $r'$  is a directed path from  $W$  to  $Y$ , and the path corresponding to  $s'$  is a directed path  $W$  to  $V$ . Hence,  $V \in \mathbf{F}_{\mathcal{D}}(\mathbf{X}, \mathbf{Y})$  ( $V \in \mathbf{F}_{\mathcal{M}}(\mathbf{X}, \mathbf{Y})$ ), so that  $\mathbf{Z}$  does not satisfy condition (a) of the AC relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{D}$  ( $\mathcal{M}$ ).  $\square$

## 5.3 PROOF OF LEMMA 5.4

We first define a distance between a path and a set in Definition 5.6. We then give the proof of Lemma 5.4. This proof relies on Lemma 5.7 and Lemma 5.8 which are given later in this section.

**Definition 5.6. (Distance-from- $\mathbf{Z}$ ; Zhang, 2006)** Given a path  $p$  from  $\mathbf{X}$  to  $\mathbf{Y}$  that is  $m$ -connecting given  $\mathbf{Z}$  in a DAG or MAG, for every collider  $Q$  on  $p$ , there is a directed path (possibly of zero length) from  $Q$  to a member of  $\mathbf{Z}$ . Define the distance-from- $\mathbf{Z}$  of  $Q$  to be the length of a shortest directed path (possibly of length 0) from  $Q$  to  $\mathbf{Z}$ , and define the distance-from- $\mathbf{Z}$  of  $p$  to be the sum of the distances from  $\mathbf{Z}$  of the colliders on  $p$ .

**Proof of Lemma 5.4:** Let  $\mathcal{G}$  represent an amenable

CPDAG (PAG) that satisfies condition (0) of the GAC relative to  $(\mathbf{X}, \mathbf{Y})$ , and let  $\mathbf{Z}$  satisfy condition (1) of the GAC relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{G}$ .

We first prove that if  $\mathbf{Z}$  does not satisfy condition (2) of the GAC relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{G}$ , then  $\mathbf{Z}$  does not satisfy condition (b) of the AC relative to  $(\mathbf{X}, \mathbf{Y})$  in any DAG (MAG) in  $[\mathcal{G}]$ . Thus, assume that there is a proper definite status non-causal path  $p$  from  $X \in \mathbf{X}$  to  $Y \in \mathbf{Y}$  that is m-connecting given  $\mathbf{Z}$  in  $\mathcal{G}$ . Consider any DAG  $\mathcal{D}$  (MAG  $\mathcal{M}$ ) in  $[\mathcal{G}]$ . Then the path corresponding to  $p$  in  $\mathcal{D}$  ( $\mathcal{M}$ ) is a proper non-causal m-connecting path from  $\mathbf{X}$  to  $\mathbf{Y}$  given  $\mathbf{Z}$ . Hence,  $\mathbf{Z}$  violates condition (b) of the AC relative to  $(\mathbf{X}, \mathbf{Y})$  and  $\mathcal{D}$  ( $\mathcal{M}$ ).

Next, we prove that if  $\mathbf{Z}$  violates condition (b) of the AC relative to  $(\mathbf{X}, \mathbf{Y})$  in some DAG (MAG) in  $[\mathcal{G}]$ , then  $\mathbf{Z}$  violates condition (2) of the GAC relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{G}$ . Thus, assume that there is a DAG  $\mathcal{D}$  (MAG  $\mathcal{M}$ ) in  $[\mathcal{G}]$  such that there is a proper non-causal m-connecting path from  $\mathbf{X}$  to  $\mathbf{Y}$  in  $\mathcal{D}$  ( $\mathcal{M}$ ) given  $\mathbf{Z}$ . We choose a shortest such path  $p$ , such that no equally short proper non-causal m-connecting path has a shorter distance-from- $\mathbf{Z}$  than  $p$ . By Lemma 5.8 below, the corresponding path  $p^*$  in  $\mathcal{G}$  is an m-connecting proper definite status non-causal path from  $\mathbf{X}$  to  $\mathbf{Y}$  given  $\mathbf{Z}$ . Hence  $\mathbf{Z}$  violates condition (b) of the GAC relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{G}$ .  $\square$

**Lemma 5.7.** *Let  $\mathcal{M}$  represent a MAG (DAG) and let  $\mathcal{P}$  be the PAG (CPDAG) of  $\mathcal{M}$ . Let  $\mathcal{P}$  satisfy condition (0) of the GAC relative to  $(\mathbf{X}, \mathbf{Y})$ , and let  $\mathbf{Z}$  satisfy condition (1) of the GAC relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{P}$ . Let  $p$  be a shortest proper non-causal path from  $\mathbf{X}$  to  $\mathbf{Y}$  that is m-connecting given  $\mathbf{Z}$  in  $\mathcal{M}$  and let  $p^*$  denote the corresponding path constituted by the same sequence of variables in  $\mathcal{P}$ . Then  $p^*$  is a proper definite status non-causal path in  $\mathcal{P}$ .*

Lemma 5.7 is related to Lemma 1 from Zhang (2006). The proof of Lemma 5.7 is given in the supplement.

**Lemma 5.8.** *Let  $\mathcal{M}$  represent a MAG (DAG) and let  $\mathcal{P}$  be the PAG (CPDAG) of  $\mathcal{M}$ . Let  $\mathcal{P}$  satisfy condition (0) of the GAC relative to  $(\mathbf{X}, \mathbf{Y})$ , and let  $\mathbf{Z}$  satisfy condition (1) of the GAC relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{P}$ . Let  $p$  be a shortest proper non-causal path from  $\mathbf{X}$  to  $\mathbf{Y}$  that is m-connecting given  $\mathbf{Z}$  in  $\mathcal{M}$ , such that no equally short such path has a shorter distance-from- $\mathbf{Z}$  than  $p$ . Let  $p^*$  denote the corresponding path constituted by the same sequence of variables in  $\mathcal{P}$ . Then  $p^*$  is a proper definite status non-causal path from  $\mathbf{X}$  to  $\mathbf{Y}$  that is m-connecting given  $\mathbf{Z}$  in  $\mathcal{P}$ .*

Lemma 5.8 is related to Lemma 2 from Zhang (2006).

**Proof of Lemma 5.8.** By Lemma 5.7,  $p^*$  is a proper definite status non-causal path in  $\mathcal{P}$ . It is only left to prove that  $p^*$  is m-connecting given  $\mathbf{Z}$  in  $\mathcal{P}$ .

Every definite non-collider on  $p^*$  in  $\mathcal{P}$  corresponds to a non-collider on  $p$  in  $\mathcal{M}$ , and every collider on  $p^*$  is also

a collider on  $p$ . Since  $p$  is m-connecting given  $\mathbf{Z}$ , no non-collider is in  $\mathbf{Z}$  and every collider has a descendant in  $\mathbf{Z}$ . Let  $Q$  be an arbitrary collider (if there is one). Then there is a directed path (possibly of zero length) from  $Q$  to a node in  $\mathbf{Z}$  in  $\mathcal{M}$ . Let  $d$  be a shortest such path from  $Q$  to a node  $Z \in \mathbf{Z}$ . Let  $d^*$  denote the corresponding path in  $\mathcal{P}$ , constituted by the same sequence of variables. Then  $d^*$  is an unshielded possibly directed path from  $Q$  to  $Z$  in  $\mathcal{P}$  (Lemma B.1 from Zhang (2008)).

It is only left to show that  $d^*$  is a directed path. If  $d^*$  is of zero length, this is trivially true. Otherwise, suppose for contradiction that there is a circle mark on  $d^*$ . Then  $d^*$  must start with a circle mark at  $Q$  (cf. Lemma B.2 from Zhang, 2008 and Lemma 7.2 from Maathuis and Colombo, 2015; see Section 1 of the supplement).

Let  $S$  be the first node on  $d$  after  $Q$ . If  $S$  is not a node on  $p$ , then following the proof of Lemma 2 from Zhang (2006) there is a path  $p' = p(X, W) \oplus W \bullet \rightarrow S \leftarrow \bullet V \oplus p(V, Y)$ , where  $W$  and  $V$  are nodes distinct from  $Q$  on  $p(X, Q)$  and  $p(Q, Y)$  respectively and  $p'$  is m-connecting given  $\mathbf{Z}$  in  $\mathcal{M}$ . Since  $p'$  is non-causal and shorter than  $p$ , or as long as  $p$  but with a shorter distance-from- $\mathbf{Z}$  than  $p$ , the path  $p'$  must be non-proper, i.e.  $S \in \mathbf{X}$ . But, in that case the path  $\langle S, V \rangle \oplus p(V, Y)$  is a proper non-causal m-connecting path from  $\mathbf{X}$  to  $\mathbf{Y}$  given  $\mathbf{Z}$  that is shorter than  $p$  in  $\mathcal{M}$ . This contradicts our assumption about  $p$ .

If  $S$  is a node on  $p$ , then it lies either on  $p(X, Q)$  or  $p(Q, Y)$ . Assume without loss of generality that  $S$  is on  $p(Q, Y)$ . Following the proof of Lemma 2 from Zhang (2006), there exists a path  $p' = p(X, W) \oplus W \bullet \rightarrow S \oplus p(S, Y)$  in  $\mathcal{M}$ , where  $W$  is a node on  $p(X, Q)$  distinct from  $Q$  that is m-connecting given  $\mathbf{Z}$  in  $\mathcal{M}$ . Since  $p'$  is proper, and shorter than  $p$ , or as long as  $p$  but with a shorter distance-from- $\mathbf{Z}$  than  $p$ , the path  $p'$  must be causal in  $\mathcal{M}$ . Let  $p'^*$  denote the corresponding path constituted by the same sequence of variables in  $\mathcal{P}$ . Then  $p'^*$  is a possibly causal path and  $Z \in \text{PossDe}(S, \mathcal{P})$ , so  $Z \in \mathbf{F}_{\mathcal{P}}(\mathbf{X}, \mathbf{Y}) \cap \mathbf{Z}$ . This is in contradiction with our assumption of  $\mathbf{Z}$  satisfying condition (1) of the GAC relative to  $(\mathbf{X}, \mathbf{Y})$  in  $\mathcal{P}$ .

Thus, the path  $d^*$  is directed and  $Q$  is an ancestor of  $\mathbf{Z}$  in  $\mathcal{P}$ . This proves that  $p^*$  is a proper definite status non-causal path from  $\mathbf{X}$  to  $\mathbf{Y}$  that is m-connecting given  $\mathbf{Z}$  in  $\mathcal{M}$ .  $\square$

## 6 DISCUSSION

We have derived a generalized adjustment criterion that is necessary and sufficient for adjustment in DAGs, MAGs, CPDAGs and PAGs. Our criterion unifies existing criteria for DAGs and MAGs, and provides a new result for CPDAGs and PAGs, where only a sufficient criterion ex-

isted until now. This is relevant in practice, in particular in combination with algorithms that can learn CPDAGs or PAGs from observational data.

Our generalized adjustment criterion is stated in terms of paths that need to be blocked, which is intuitively appealing. A logical next step for future research would be to transform our criterion into an algorithmically constructive version that could be used to efficiently perform tasks like enumeration of all minimal adjustment sets for a given graph. This has already been done for DAGs and MAGs by van der Zander et al. (2014), and we strongly suspect that their results can be extended to CPDAGs and PAGs as well. In a similar spirit, it would be desirable to have an easily checkable condition to determine if there exists any adjustment set at all, as done for the generalized back-door criterion for single interventions by Maathuis and Colombo (2015). In turn, these results could then be used to characterize distances between graphs, as done by Peters and Bühlmann (2015). Future work might also explore under which circumstances our restriction to not allow for latent selection variables might be relaxed, or whether our criterion could be combined with methods to recover from selection bias (Bareinboim et al., 2014).

As pointed out in Section 4, our criterion sometimes has to interpret PAGs or MAGs differently than DAGs or CPDAGs. This is the case precisely when the first edge on some proper possibly causal path in a MAG or PAG is not visible. However, this difference in interpretation is irrelevant for DAGs or CPDAGs that would be amenable when viewed as a MAG or PAG. For instance, if we are given a DAG  $\mathcal{D}$  that is amenable when interpreted as a MAG  $\mathcal{M}$ , then its adjustment sets also work for every DAG that the MAG  $\mathcal{M}$  represents, many of which could contain latent confounding variables. Reading a DAG as a MAG (or a CPDAG as a PAG) can thus allow computing adjustment sets that are to some extent invariant to confounding.

We note that an adjustment set relative to  $(\mathbf{X}, \mathbf{Y})$  in a given graph can only exist if the total causal effect of  $\mathbf{X}$  on  $\mathbf{Y}$  is identifiable in the graph. If the effect of  $\mathbf{X}$  on  $\mathbf{Y}$  is not identifiable, one may be interested in computing all possible total causal effects of  $\mathbf{X}$  on  $\mathbf{Y}$  for DAGs represented by the given graph. Such an approach is used in the IDA algorithm of Maathuis et al. (2009, 2010), by considering all DAGs represented by a CPDAG and applying back-door adjustment to each of these DAGs. Similar ideas could be used for MAGs and PAGs, but listing all relevant DAGs described by a MAG or PAG seems rather non-trivial.

There is also an interesting connection between amenability and instrumental variables: a MAG or PAG  $\mathcal{G}$  with  $\mathbf{X} = \{X\}$  is amenable with respect to  $(\mathbf{X}, \mathbf{Y})$  whenever it contains an *instrument*  $I$ , i.e. there exists a variable that is a parent of  $X$  but not a parent of any child of  $X$  (e.g.,  $I$  in Figure 1a). Thus, instruments are useful to find adjustment

sets in nonparametric graphical models that allow for latent confounding. This connection is perhaps surprising given that the notion of instruments originates from causal effect identifications in linear models (Angrist et al., 1996).

In summary, our generalized adjustment criterion exhaustively characterizes the options to identify total causal effects by covariate adjustment in DAGs, MAGs, CPDAGs, and PAGs. Our results entail several existing, less general or less powerful ones (Pearl, 1993; Shpitser et al., 2012; Textor and Liškiewicz, 2011; van der Zander et al., 2014; Maathuis and Colombo, 2015) as special cases.

## Acknowledgements

This research was supported by the Swiss National Science Foundation (200021\_149760).

## References

- Ali, R. A., Richardson, T. S., and Spirtes, P. (2009). Markov equivalence for ancestral graphs. *Ann. Stat.*, 37:2808–2837.
- Angrist, J. D., Imbens, G. W., and Rubin, D. B. (1996). Identification of causal effects using instrumental variables. *J. Am. Stat. Assoc.*, 91(434):444–455.
- Bareinboim, E., Tian, J., and Pearl, J. (2014). Recovering from selection bias in causal and statistical inference. In *Proceedings of AAAI 2014*, pages 2410–2416.
- Chickering, D. M. (2003). Optimal structure identification with greedy search. *J. Mach. Learn. Res.*, 3:507–554.
- Claassen, T., Mooij, J., and Heskes, T. (2013). Learning sparse causal models is not NP-hard. In *Proceedings of UAI 2013*, pages 172–181.
- Colombo, D. and Maathuis, M. H. (2014). Order-independent constraint-based causal structure learning. *J. Mach. Learn. Res.*, 15:3741–3782.
- Colombo, D., Maathuis, M. H., Kalisch, M., and Richardson, T. S. (2012). Learning high-dimensional directed acyclic graphs with latent and selection variables. *Ann. Stat.*, 40:294–321.
- Maathuis, M. H. and Colombo, D. (2015). A generalized back-door criterion. *Ann. Stat.*, 43:1060–1088.
- Maathuis, M. H., Colombo, D., Kalisch, M., and Bühlmann, P. (2010). Predicting causal effects in large-scale systems from observational data. *Nat. Methods*, 7:247–248.
- Maathuis, M. H., Kalisch, M., and Bühlmann, P. (2009). Estimating high-dimensional intervention effects from observational data. *Ann. Stat.*, 37:3133–3164.

- Meek, C. (1995). Causal inference and causal explanation with background knowledge. In *Proceedings of UAI 1995*, pages 403–410.
- Pearl, J. (1993). Comment: Graphical models, causality and intervention. *Stat. Sci.*, 8:266–269.
- Pearl, J. (2009). *Causality*. Cambridge University Press, Cambridge, second edition.
- Peters, J. and Bühlmann, P. (2015). Structural intervention distance (SID) for evaluating causal graphs. *Neural Comput.*, 27:771–799.
- Richardson, T. and Spirtes, P. (2002). Ancestral graph Markov models. *Ann. Stat.*, 30:962–1030.
- Robins, J. (1986). A new approach to causal inference in mortality studies with a sustained exposure period—application to control of the healthy worker survivor effect. *Math. Mod.*, 7:1393–1512.
- Rubin, D. (2008). Author’s reply. *Stat. Med.*, 27:2741–2742.
- Shpitser, I. (2012). Appendix to “On the validity of covariate adjustment for estimating causal effects”. Unpublished manuscript.
- Shpitser, I. and Pearl, J. (2006). Identification of joint interventional distributions in recursive semi-markovian causal models. In *Proceedings of AAAI 2006*, pages 1219–1226.
- Shpitser, I., VanderWeele, T., and Robins, J. M. (2012). On the validity of covariate adjustment for estimating causal effects. In *Proceedings of UAI 2010*, pages 907–916.
- Shrier, I. (2008). Letter to the editor. *Stat. Med.*, 27:2740–2741.
- Shrier, I. and Platt, R. W. (2008). Reducing bias through directed acyclic graphs. *BMC Med. Res. Methodol.*, 8(70).
- Spirtes, P., Glymour, C., and Scheines, R. (2000). *Causation, Prediction, and Search*. MIT Press, Cambridge, second edition.
- Textor, J. and Liškiewicz, M. (2011). Adjustment criteria in causal diagrams: An algorithmic perspective. In *Proceedings of UAI 2011*, pages 681–688.
- Tian, J. and Pearl, J. (2002). A general identification condition for causal effects. In *Proceedings of AAAI 2002*, pages 567–573.
- van der Zander, B., Liškiewicz, M., and Textor, J. (2014). Constructing separators and adjustment sets in ancestral graphs. In *Proceedings of UAI 2014*, pages 907–916.
- West, S. G. and Koch, T. (2014). Restoring causal analysis to structural equation modeling. *Struct. Equ. Modeling*, 21:161–166.
- Westreich, D. and Greenland, S. (2013). The table 2 fallacy: presenting and interpreting confounder and modifier coefficients. *Am. J. Epidemiol.*, 177:292–298.
- Zhang, J. (2006). *Causal Inference and Reasoning in Causally Insufficient Systems*. PhD thesis, Carnegie Mellon University.
- Zhang, J. (2008). On the completeness of orientation rules for causal discovery in the presence of latent confounders and selection bias. *Artif. Intell.*, 172:1873–1896.

---

# Optimal Threshold Control for Energy Arbitrage with Degradable Battery Storage

---

**Marek Petrik**

IBM T. J. Watson Research Center  
Yorktown, NY 10598  
mpetrik@us.ibm.com

**Xiaojian Wu**

School of Computer Science  
University of Massachusetts Amherst  
Amherst, MA 01003  
xiaojian@cs.umass.edu

## Abstract

Energy arbitrage has the potential to make electric grids more efficient and reliable. Batteries hold great promise for energy storage in arbitrage but can degrade rapidly with use. In this paper, we analyze the impact of storage degradation on the structure of optimal policies in energy arbitrage. We derive properties of the battery degradation response that are sufficient for the existence of optimal threshold policies, which are easy to interpret and compute. Our experimental results suggest that explicitly considering battery degradation in optimizing energy arbitrage significantly improves solution quality.

## 1 INTRODUCTION

Energy storage and arbitrage play an important role in numerous domains including hybrid vehicle propulsion or electric grids [Walawalkar et al., 2007]. An important challenge in this domain is to decide how much energy to store or release based on current and expected future prices. Related resource efficiency optimization problems have been recently studied, among others, in the computational sustainability research area [Gomes, 2009, Petrik and Zilberstein, 2011, Ermon et al., 2011, Ermon et al., 2013]. Since computing an energy arbitrage policy is a difficult sequential *stochastic* optimization problem subject to significant uncertainty, it is particularly relevant to the computational sustainability community.

In this paper, we are concerned with optimizing energy arbitrage under stochastically varying energy prices. The goal of the decision maker is to maximize profits by charging an energy storage device when the price of energy is low and discharging it when it is high. We show that optimal policies have a threshold structure even when *battery degradation* is considered and use this structure to develop a practical algorithm.

Our optimal threshold policy has two price-dependent thresholds  $l$  and  $u$  ( $l \leq u$ ). If the current state of charge is less than  $l$ , then the battery is charged up to  $l$ . If the current state of charge is greater than  $u$ , then the battery is discharged to  $u$ . For any state of charge between  $l$  and  $u$ , no action is taken. Such threshold structure makes charging policies easy to compute, analyze, and interpret.

The structure of optimal policies in energy arbitrage has been analyzed previously without accounting for battery degradation [Lifshitz and Weiss, 2014, Nadarajah, 2014, Van de Ven et al., 2011, Harsha and Dahleh, 2011]. In case of batteries—an important energy storage device—the degradation can be significant and often represents an important limitation due to the high cost of batteries.

Batteries degrade most noticeably by losing capacity to hold charge. Modeling how usage patterns affect different battery types (e.g. NiMH and Li-ion) is an important research problem [Ramadass et al., 2003, Aurbach, 2000]. In Li-ion batteries, the degradation is predominantly influenced by 1) the state of charge, 2) the rate of charge and discharge, and 3) the ambient temperature. We aim, in this work, to minimize the battery’s capacity loss as it is influenced by the state of charge.

Battery degradation has been explicitly considered in optimizing energy storage in hybrid vehicles [Bashash et al., 2011, Serrao et al., 2005, Hoke et al., 2011, Moura et al., 2011]. These methods solve a discretized dynamic program. The drawback of this approach is that the computed policies are complex, hard to implement and interpret. In addition, the discretization and sampling issues can significantly degrade solution quality as we show experimentally. On the other hand, the optimality of threshold policies has been studied widely in the inventory management literature [Porteus, 2002]. There is, however, no concept of storage degradation in traditional inventory management domains.

The existence of an optimal threshold policy in our setting is somewhat surprising. The standard threshold policy results rely on the fact that the optimal value function is convex or  $k$ -convex. The optimal value function for non-trivial battery degradation functions, as we show, may be

non-convex. Yet, we establish the existence of a threshold policy through the convexity of an auxiliary optimization function.

The remainder of the paper is organized as follows. Section 2 describes the overall arbitrage model. We primarily study models that faithfully capture the energy arbitrage setting in electric grids. Section 3 describes the model of battery degradation and derives some basic properties of the degradation function. Then, we show sufficient conditions for the existence of threshold policies in Section 4 and describe how the structure can be used in computing an optimized policy in Section 5. Section 6 describes an application of the methodology with the analysis of the results and a comparison to discretization-based methods.

## 2 ENERGY STORAGE MODEL

This section describes the model of energy arbitrage and storage. We assume multiple finite *known* price levels and a stochastic evolution given a limited storage capacity. In particular, the storage is assumed to be an electrical battery that degrades when energy is stored or retrieved.

The underlying model is a Markov decision process. We assume a discrete-time problem with either a finite horizon  $T$  or a discounted infinite horizon. Prices are governed by a Markov process with states  $\Theta$ . There are two energy prices in each time step:  $p^i : \Theta \rightarrow \mathbb{R}_+$  is the purchase (or input) price and  $p^o : \Theta \rightarrow \mathbb{R}_+$  is the selling (or output) price. To simplify notation, the difference in these prices is also used to model the energy loss in the charging and discharging processes. In other words, the prices measure the cost of energy as added or subtracted from the storage and may not actually be sold or purchased.

We use  $s$  to denote the available battery capacity with  $s_0$  denoting the initial capacity. The current state of charge is denoted as  $x$  or  $y$  and must satisfy that  $0 \leq x_t \leq s_t$  at any time step  $t$ . The action is the amount of energy to charge or discharge, which is denoted by  $u$ . Positive  $u$  indicates that energy is purchased to charge the battery; negative  $u$  indicates the sale of energy.

We will make the following assumption regarding the purchase and selling prices.

**Assumption 1.** Purchase price is higher than the selling price per unit in a time step:

$$p_\theta^i \geq p_\theta^o \quad \forall \theta \in \Theta.$$

Assumption 1 is virtually always satisfied in practice. If violated, direct arbitrage by simultaneously purchasing and selling energy in a single time step would then equalize the prices. In addition, the purchase price  $p^i$  will be greater than the selling price  $p^o$  due to the inefficiencies involved in charging and discharging.

As mentioned above, the focus of the paper is on degradation of battery capacity as a function of its use. In particular, we model the degradation as a function of the battery

capacity when charged or discharged. We use a general model of battery degradation with a specific focus on Li-ion batteries. The degradation function  $d(x, u) \in \mathbb{R}_+$  represents the battery capacity loss after starting at the state of charge  $x \geq 0$  and charging (discharging if negative) by  $u$  with  $-x \leq u \leq s_0 - x$ . This function indicates the loss of capacity, such that:

$$s_{t+1} = s_t - d(x_t, u_t)$$

We discuss the degradation function in more detail in Section 3.

Our model makes several simplifying assumptions that are reasonable in an electric grid scenario, but may not apply to other scenarios such as a hybrid vehicle battery storage. In particular, we assume that the purchase and selling prices are independent of the energy quantity sold or purchased and battery degradation is independent of current and temperature.

The state set in the Markov decision problem is composed of  $(x, s, \theta)$  where  $x$  is the state of charge,  $s$  is the battery capacity, and  $\theta \in \Theta$  is the state of the price process. The available actions in a state  $(x, s, \theta)$  are  $u$  such that  $-x \leq u \leq s - x$ . The transition is from  $(x_t, s_t, \theta_t)$  to  $(x_{t+1}, s_{t+1}, \theta_{t+1})$  given action  $u_t$  is:

$$\begin{aligned} x_{t+1} &= x_t + u_t \\ s_{t+1} &= s_t - d(x_t, u_t) \end{aligned}$$

The probability of this transition is given by  $\mathbb{P}[\theta_{t+1}|\theta_t]$ . The reward for this transition is:

$$r((x_t, s_t, \theta_t), u_t) = \begin{cases} -u_t \cdot p^i - c^d \cdot d(x_t, u_t) & \text{if } u_t \geq 0 \\ -u_t \cdot p^o - c^d \cdot d(x_t, u_t) & \text{if } u_t < 0 \end{cases}$$

That is, the reward captures the monetary value of the transaction minus a penalty for degradation of the battery. Here,  $c^d$  represents the cost of a unit of lost battery capacity.

The solution of the Markov decision process is a policy  $\pi$ , which can be computed from a value function  $v$  and a post-decision (or state-action) value function  $q$ . We focus on both the discounted infinite horizon with a discount factor  $\lambda \in (0, 1)$  and the finite horizon with the undiscounted total return criterion.

The Bellman optimality equations for this problem are:

$$\begin{aligned} q_T(x, s, \theta_T) &= 0 \\ v_t(x, s, \theta_t) &= \min\{p_{\theta_t}^i [u]_+ + p_{\theta_t}^o [u]_- + \\ &\quad + c^d d(x, u) + \\ &\quad + q_t(x + u, s - d(x, u), \theta_t) : \\ &\quad : u \in [-x, s - x]\} \\ q_t(x, s, \theta_t) &= \lambda \cdot \mathbb{E}[v_{t+1}(x, s, \theta_{t+1})] \end{aligned} \quad (2.1)$$

where  $[u]_+ = \max\{u, 0\}$  and  $[u]_- = \min\{u, 0\}$  and the expectation is taken over  $P(\theta_{t+1}|\theta_t)$ . For finite horizon case,  $\lambda = 1$ .

For the purpose of our theoretical analysis, we assume that the lost capacity is immediately replaced and therefore the battery capacity does not actually change ( $s_{t+1} = s_t = s_0$ ). Instead, the degradation induces a penalty in the form of capacity replacement cost governed by  $c_d$ . In that case, the capacity  $s$  can be omitted from the definition of  $v$  in (2.1). The experimental results, however, study the setting in which the capacity is not immediately replaced.

### 3 BATTERY DEGRADATION FUNCTION

This section describes properties of the degradation function  $d(x, u)$  that can capture the behavior of Li-ion batteries [Aurbach, 2000, Ramadass et al., 2003] and can guarantee the existence of an optimal threshold policy. We focus on Li-ion batteries because of their ubiquity and considerable promise in future applications [Peterson et al., 2010]. The chemical processes in other battery types—such as NiMH and NiCd—are often quite different [Serrao et al., 2005].

As noted above, we consider the dependence of the degradation only on the state of charge because the other variables, such as the temperature and the current, can be efficiently controlled in an electric grid energy application.

In broad terms, a Li-ion battery degrades significantly while the state of charge is either very low or very high. A naive policy that minimizes battery degradation will therefore attempt to use the battery as close to approximately 50% state of charge as possible.

Instead of considering a single degradation function, we define a class of functions  $d(x, u)$ , *continuous* in  $u$  for  $x \in [0, s_0]$ , that in addition satisfy the following properties.

- A1 *Convexity*: function  $d(x, u)$  is convex in  $u$  for all  $x \in [0, s_0]$ .
- A2 *Memorylessness*:  $d(x, u_1 + u_2) = d(x, u_1) + d(x, u_2)$  when  $\text{sgn}(u_1) = \text{sgn}(u_2)$ .
- A3 *Cycle linearity*: function  $d(0, u) + d(u, -u)$  is linear in  $u$ .

Next we informally describe the meaning of the properties above; a more detailed analysis of functions that satisfy these properties follows later. The property A1 is generally satisfied in Li-ion batteries in which the degradation is more severe near the extreme range of the state of charge. The property A2 requires that the degradation due to the charging is independent of the amount charged, but instead depends only on the current state of charge. Finally, the property A3 requires that the degradation due to charging an arbitrary amount from the state of charge 0 and subsequently fully discharging is linear in the amount charged.

The assumptions above, unfortunately, are hard to grasp intuitively and therefore difficult to justify. To elucidate the definitions, consider the alternative definition of

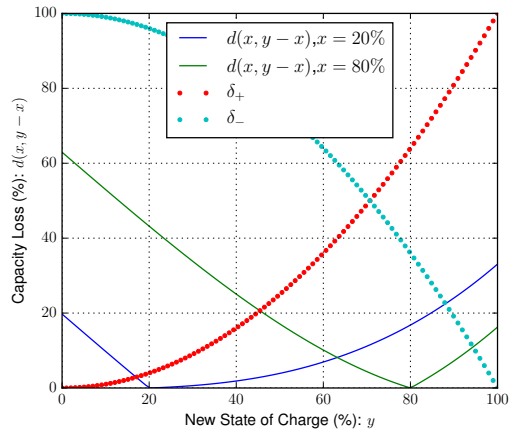


Figure 1: Example degradation function with  $\delta_+(x) = x^2$  and  $\delta_-(x) = 1 - x^2$ .

$d(x, u)$  based on the degradation at any state of charge for an infinitesimally small amount of energy charged  $\delta_+ : [0, s_0] \rightarrow \mathbb{R}_+$  or discharged  $\delta_- : [0, s_0] \rightarrow \mathbb{R}_+$ . The degradation function is then simply defined as the following integral:

$$d(x, u) = \begin{cases} \int_x^{x+u} \delta_+(y) dy & \text{if } u \geq 0 \\ \int_{x+u}^x \delta_-(y) dy & \text{if } u < 0 \end{cases}. \quad (3.1)$$

Fig. 1 depicts an example of the immediate degradation functions  $\delta_+$  and  $\delta_-$  and the corresponding degradation function  $d$  for two values of the current state of charge.

The following proposition describes how the properties of  $\delta_+$  and  $\delta_-$  translate to properties of  $d(x, u)$ .

**Proposition 3.1.** *When  $d$  is defined as in (3.1) and*

- (i) *both  $\delta_-$  and  $\delta_+$  are continuous on  $[0, s_0]$*
- (ii)  *$\delta_+$  is nondecreasing and  $\delta_-$  is nonincreasing*
- (iii)  *$\delta_+(y) + \delta_-(y)$  is a constant for any  $y \in [0, s_0]$*

*Then, the battery degradation function  $d(x, u)$  satisfies the properties of A1, A2 and A3.*

*Proof.* The property (i) implies that there exist anti-derivatives  $D_+$  and  $D_-$  to  $\delta_+$  and  $\delta_-$  on the appropriate intervals. Then:

$$d(x, u) = \begin{cases} D_+(x+u) - D_+(x) & \text{if } u \geq 0 \\ D_-(x) - D_-(x+u) & \text{if } u < 0 \end{cases}$$

We prove each property individually.

A1: *Convexity.* The convexity for a fixed  $x$  and  $u \neq 0$  follows directly from the convexity of  $D_+$  and  $-D_-$ . The functions  $D_+(x, u)$  and  $D_-(x, u)$  are convex because their first derivatives  $\delta_+$  and  $-\delta_-$  are nondecreasing [Boyd and Vandenberghe, 2004]. Since both  $D_+$  and  $D_-$  are non-negative functions and  $d(x, 0) = 0$ , we have



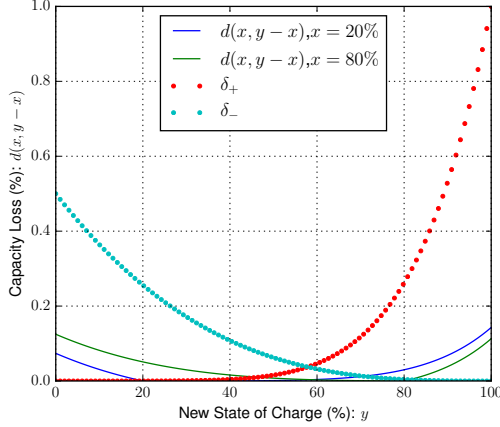


Figure 2: Example degradation function with  $\delta_+(x) = 0.01 \cdot x^6$  and  $\delta_-(x) = 0.005 \cdot (1-x)^3$ .

that  $u = 0$  is the minimum of  $d(x, u)$  and therefore it satisfies the convexity condition.

A2: Memorylessness. If  $u_1 > 0$  and  $u_2 > 0$ ,

$$\begin{aligned} d(x, u_1 + u_2) &= \int_x^{x+u_1+u_2} \delta_+(y) dy \\ &= \int_x^{x+u_1} \delta_+(y) dy + \int_{x+u_1}^{x+u_1+u_2} \delta_+(y) dy \\ &= d(x, u_1) + d(x + u_1, u_2). \end{aligned}$$

The property holds similarly when  $u_1 < 0$  and  $u_2 < 0$ .

A3: Cycle linearity. Let  $u > 0$ ,

$$d(0, u) + d(u, -u) = \int_0^u \delta_+(y) dy + \int_0^u \delta_-(y) dy$$

since  $\delta_+(y) + \delta_-(y)$  is a constant, the cycle linearity is satisfied.  $\square$

Note that the degradation function in Fig. 1 satisfies the properties A1–A3. This can be easily seen since  $\delta_+(x) + \delta_-(x) = 1$ . However, the degradation of Li-ion batteries may not always satisfies these properties. Fig. 2 illustrates an example, based on real Li-ion behavior [Bashash et al., 2011], which violates A3. The precise form of the degradation function for any particular battery design can be obtained either experimentally or by simulation [Ramadesigan et al., 2012].

## 4 STRUCTURE OF OPTIMAL POLICIES

In this section, we show the existence of an optimal threshold policy in the battery storage problem if properties A1–A3 are satisfied. The analysis is based on a finite-horizon version of the problem and we discuss how this structure generalizes to discounted infinite horizon problems later in the section.

A two-threshold charge policy is defined as follows:



Figure 3: Example of a threshold policy. The upper (red) line represents  $C(\theta)$ ; the lower (green) line represents  $c(\theta)$ .

**Definition 4.1.** A two-threshold charge policy with thresholds  $(c_{t,\theta}, C_{t,\theta})$  with  $c_{t,\theta} \leq C_{t,\theta}$  for some  $\theta \in \Theta$  and  $t = 1 \dots T$  is defined as:

$$u_t = \begin{cases} c_{t,\theta} - x_t & \text{when } x_t \leq c_{t,\theta} \\ C_{t,\theta} - x_t & \text{when } x_t \geq C_{t,\theta} \\ 0 & \text{otherwise} \end{cases} \quad (4.1)$$

where  $t$  is the current time step,  $x_t$  is the current battery charge,  $\theta_t$  is the price level state, and  $u_t$  is the change in the state of charge.

One of the main appealing properties of a threshold policy is its simplicity and interpretability. Fig. 3 depicts an example of a threshold policy. The  $x$ -axis represents the state of the price process  $\theta$ . In this example, the price of energy grows linearly with  $\theta$  and the price transitions behave as a martingale. If the current state of charge is in the red region, the next step is to discharge the battery to the red line. Similarly, states in the green region are charged up to the green line.

Note that the policy in Fig. 3 behaves very intuitively. When the price of energy is low (small  $\theta$ ), the battery is charged to a high level. It is not charged fully, however, to prevent excessive degradation of capacity. No action is taken for the medium energy price. When the energy price increases to its maximum level, the battery is fully discharged.

We are now ready to state the main theoretical result of the paper.

**Theorem 4.2.** Assume that the battery degradation function satisfies properties A1, A2, and A3 and  $\lambda = 1$ . Then, there exists an optimal two-threshold charge policy for the finite horizon problem with some time-dependent thresholds  $(c(t, \theta), C(t, \theta))$ .

To prove the theorem, we need to show several auxiliary properties. The following lemma describes the properties of the degradation function induced by the assumptions above.

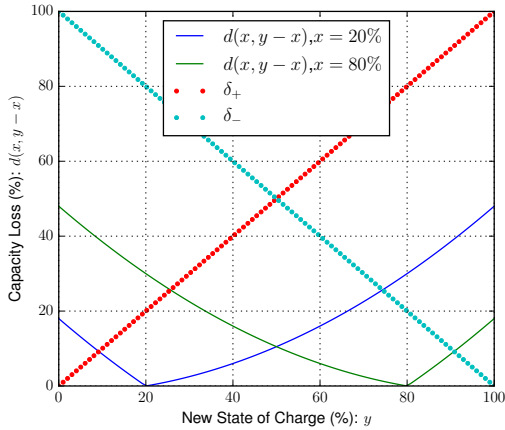


Figure 4: Degradation function in Example 4.4.

**Lemma 4.3.** A degradation function  $d$  that satisfies property A2 also satisfies:

- (i)  $d(x, 0) = 0$
- (ii)  $d(x, y - x) = d(0, y) - d(0, x)$  when  $y \geq x$
- (iii)  $d(x, y - x) = d(x, -x) - d(y, -y)$  when  $y \leq x$

In addition, when  $d$  satisfies property A1, then:

- (iv)  $d(x, -x)$  is concave in  $x$
- (v)  $d(x, y - x) = \max\{d(0, y) - d(0, x), d(x, -x) - d(y, -y)\}$

*Proof.* The lemma follows by simple algebraic manipulation for the individual cases as follows.

Case (i):

$$d(x, u + 0) = d(x, 0) + d(x + 0, u).$$

Case (ii):

$$d(0, y) = d(0, x + (y - x)) = d(0, x) + d(x, y - x).$$

Case (iii):

$$d(x, -x) = d(x, (y - x) - y) = d(x, y - x) + d(y, -y).$$

Case (iv): by rewriting  $d(s_0, -s_0) = d(s_0, -(s_0 - x) - x)$  we get:

$$d(x, -x) = d(s_0, -s_0) - d(s_0, x - s_0).$$

Function  $d(x, -x)$  is concave because  $d(s_0, x - s_0)$  is convex.

Case (v): Note that  $d(0, x)$  is non-decreasing, and  $d(x, -x)$  is non-increasing. The proof then readily follows from properties (ii) and (iii).  $\square$

We are now ready to prove the main result. The typical proof of the threshold property in inventory management settings is based on convexity of the value function. Unfortunately, as the example described below in Example 4.4

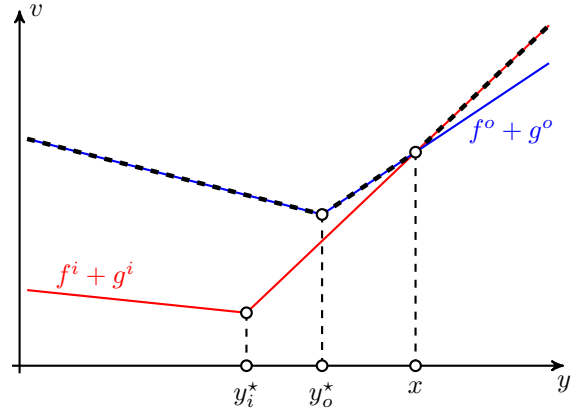


Figure 5: Example of functions in the proof of Theorem 4.2. The current charge is  $x$  and the optimal action is to discharge to  $y_o^*$ .

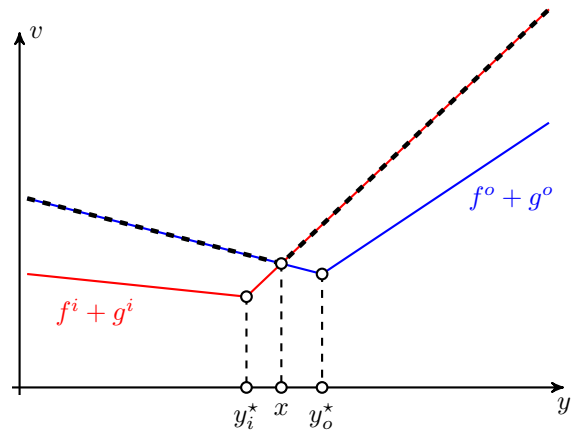


Figure 6: Example of functions in the proof of Theorem 4.2. The current charge is  $x$  and the optimal action is to leave the charge unchanged.

demonstrates, the value function in our setting may be non-convex. Instead, we use property A3 to establish the convexity of the post-decision value function (the concept is similar to  $q$ -function in reinforcement learning). In particular, the linearity required by A1 can be used to cancel out the value function non-convexities due to the battery degradation.

*Proof of Theorem 4.2.* Like most structural proofs in dynamic programming, the proof is based a backward induction on time steps. But first, we need to derive a more convenient representation of the optimality equation (2.1). To that effect, rewrite the term  $p_\theta^i [u]_+ + p_\theta^o [u]_-$  in (2.1) as a maximum over two functions:

$$p_\theta^i [u]_+ + p_\theta^o [u]_- = \max\{p_\theta^i u, p_\theta^o u\}.$$

This holds from Assumption 1. It will be more convenient to change the optimization variable in (2.1) from the charge difference  $u$  to the new state of charge  $y = x + u$ . Also using property (v) from Lemma 4.3 to express the degradation function, the optimality expression for  $v_t(x, \theta)$  now reads as:

$$\min_{y \in [0, s_0]} \max \left\{ p_\theta^i(y - x), p_\theta^o(y - x) \right\} + q_t(y, \theta) + c^d \cdot \max \left\{ d(0, y) - d(0, x), d(x, -x) - d(y, -y) \right\}.$$

Properties (ii) and (iii) from Lemma 4.3 and Assumption 1 imply that the two max operators attain their respective first terms if and only if  $y \geq x$  and therefore can be merged:

$$\min_{y \in [0, s_0]} \max \left\{ p_\theta^i(y - x) + c^d(d(0, y) - d(0, x)) + q_t(y, \theta), p_\theta^o(y - x) + c^d(d(x, -x) - d(y, -y)) + q_t(y, \theta) \right\}$$

Then we replace  $\max\{a, b\}$  by  $\max_{\xi \in [0, 1]} \{\xi a, (1 - \xi)b\}$ . Thus the Bellman optimality conditions become:

$$v_t(x, \theta) = \min_{y \in [0, s_0]} \max_{\xi \in [0, 1]} \phi_t(y, \xi, x, \theta),$$

where the main objective function  $\phi$  is defined as:

$$\phi_t(y, \xi, x, \theta) = \xi(f_t^i(y, \theta) + g_t^i(x, \theta)) + (1 - \xi)(f_t^o(y, \theta) + g_t^o(x, \theta)).$$

The functions  $g^i$  and  $g^o$  represent terms that are constant with respect to the optimization variable  $y$ :

$$g_t^i(x, \theta) = -p_\theta^i x - c^d d(0, x) \\ g_t^o(x, \theta) = -p_\theta^o x + c^d d(x, -x).$$

On the other hand, the functions  $f^i$  and  $f^o$  represent terms that depend on the optimization variable  $y$ :

$$f_t^i(y, \theta) = p_\theta^i y + c^d d(0, y) + q_t(y, \theta) \\ f_t^o(y, \theta) = p_\theta^o y - c^d d(y, -y) + q_t(y, \theta).$$

The remainder of the proof focuses on showing that  $f^i$  and  $f^o$  are convex and using this convexity to show the optimality of a threshold policy in computing the minimization over  $y$ .

We next show by induction on  $t$  from  $t = T$  to  $t = 0$  that the functions  $f_t^i$  and  $f_t^o$  are *convex* even if  $q_t$  is not. To prove the base case  $t = T$  recall that  $q_T = 0$ . The convexity of  $f_t^i$  and  $f_t^o$  then follows from property (iv) in Lemma 4.3.

To prove the inductive step, assume that the functions  $f_{t+1}^i$  and  $f_{t+1}^o$  are *convex*. Our focus is on showing convexity of  $f_t^i$ ; the derivation of convexity of  $f_t^o$  is analogous. Recall that:

$$f_t^i(y, \theta) = p_\theta^i y + c^d d(0, y) + \mathbb{E}[v_{t+1}(x, \theta_{t+1})]$$

Since given a fixed  $x$ , the function  $\phi(y, \xi, x)$  is convex-concave, continuous, and optimized on convex compact sets we have by a generalized minimax theorem (e.g. [Sion, 1958]) and further algebraic manipulation that:

$$v_{t+1}(x, \theta) = \min_{y \in [0, s_0]} \max_{\xi \in [0, 1]} \phi_{t+1}(y, \xi, x, \theta) \\ = \max_{\xi \in [0, 1]} \min_{y \in [0, s_0]} \phi_{t+1}(y, \xi, x, \theta) \\ = \max \left\{ g_{t+1}^i(x, \theta) + \min_{y \in [0, s_0]} f_{t+1}^i(y, \theta), g_{t+1}^o(x, \theta) + \min_{y \in [0, s_0]} f_{t+1}^o(y, \theta) \right\},$$

Note that the functions  $g_{t+1}^i$  and  $g_{t+1}^o$  may not be convex, but taking the maximum outside of the minimization will help to establish the convexity of  $f_t^i$ .

Next, plug in the above expression for  $v_{t+1}$  and the definitions of  $g_{t+1}^i$  and  $g_{t+1}^o$  to  $f_t^i(y, \theta)$ . Further algebraic simplification yields:

$$f_t^i(y, \theta) = c^d d(0, y) + \mathbb{E} \left[ \max \left\{ -c^d d(0, y) + L_1(y), c^d d(y, -y) + L_2(y) \right\} \right] \\ = \mathbb{E} \left[ \max \left\{ L_3(y), c^d d(0, y) + c^d d(y, -y) + L_4(y) \right\} \right]$$

where  $L_1(y) \dots L_4(y)$  represent functions that are linear or constant in  $y$ . Now, recall from Lemma 4.3 that the function  $d(y, -y)$  is concave. However, Assumption A3 implies that  $d(0, y) + d(y, -y)$  is a linear (convex) term. Point-wise maximization and expectation preserve convexity and thus the function  $f_t^i$  is convex. The analogous proof for  $f_t^o$  requires that  $d(0, y) + d(y, -y)$  is concave (or linear) and thus the linearity required by A3.

The final step in the proof is to show the two-threshold structure in the solution to the action optimization problem:

$$\min_{y \in [0, s_0]} \max \left\{ f_t^i(y, \theta) + g_t^i(x, \theta), f_t^o(y, \theta) + g_t^o(x, \theta) \right\}. \quad (4.2)$$

Figs. 5 and 6 depict examples of functions in (4.2) for two different values of the current state of charge  $x$ . The horizontal axis represents the next state of charge  $y$  and the bold dashed line highlights the maximum of the two functions. Note that the change in the current state of charge  $x$  only shifts the two functions without changing their shape.

Now, recall that from Lemma 4.3 and from the construction of (4.2), the following inequalities hold:

$$y > x \Rightarrow f_t^i(y, \theta) + g_t^i(x, \theta) \geq f_t^o(y, \theta) + g_t^o(x, \theta) \\ y < x \Rightarrow f_t^i(y, \theta) + g_t^i(x, \theta) \leq f_t^o(y, \theta) + g_t^o(x, \theta) \\ y = x \Rightarrow f_t^i(y, \theta) + g_t^i(x, \theta) = f_t^o(y, \theta) + g_t^o(x, \theta)$$

Now let  $e_i(y) = f_t^i(y, \theta) + g_t^i(x, \theta)$  and  $y_i^* \in \arg \min_{y \in [0, s_0]} e_i(y)$ . Also let  $e_o(y) = f_t^o(y, \theta) + g_t^o(x, \theta)$

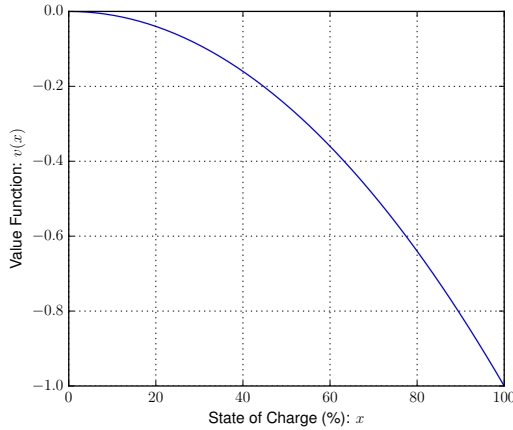


Figure 7: Value function at time  $t = 1$ .

and  $y_o^* \in \arg \min_{y \in [0, s_0]} e_o(y)$ . We will show below that the optimal solution to (4.2) can be either at  $y_i^*$ ,  $y_o^*$ , or at  $x$ . There are three possible cases:

1. If  $y_i^* > x$  then  $y_i^*$  is optimal in (4.2) because  $f_i^i(y, \theta) + g_i^i(x, \theta) \leq f_i^o(y, \theta) + g_i^o(x, \theta)$  for any  $y < x$ .
2. If  $y_o^* < x$  then  $y_o^*$  is optimal in (4.2) as above.
3. If  $y_i^* \leq x$  and  $y_o^* \geq x$  then  $x$  is optimal in (4.2). Barring the trivial case  $y_i^* = x$ , for any  $y > x$ , we have  $\frac{e_i(y) - e_i(x)}{y - x} \geq \frac{e_i(x) - e_i(y_i^*)}{x - y_i^*} \geq 0$  since  $e_i(y)$  is convex. Therefore  $e_i(y) \geq e_i(x)$  for any  $y > x$ . A similar argument for any  $y < x$  shows that  $e_o(y) \geq e_o(x)$  which proves the desired optimality of  $x$  in (4.2).

These cases correspond to the threshold policy as described in Definition 4.1.  $\square$

The following example illustrates that even when the conditions A1–A3 are satisfied, the optimal value function may not be convex.

**Example 4.4.** Consider a two stage problem with the sale price  $p^i = p^o = 2$ , the degradation cost  $c^d = 1$  and the degradation function being:

$$d(x, u) = \max\{(x + u)^2 - x^2, (s_0 - x - u)^2 - (s_0 - x)^2\}.$$

The degradation function at  $x = 0.2$  and  $x = 0.8$  is depicted in Fig. 4.

The battery capacity is  $s_0 = 1$ . It is easy to show that the optimal policy in the last stage is simply to fully discharge the battery. Then  $v_1(x) = -p^o \cdot x + c^d \cdot d(x, -x)$ . By Lemma 4.3, the function  $v_1$  is concave as illustrated in Fig. 7.

Note that the degradation function in Example 4.4 satisfies the property A3 and, therefore, this property is insufficient

to guarantee the convexity of a value function. In fact, the value function will only be convex only if  $d(x, -x)$  is linear—the degradation function is linear only when the degradation is independent of the battery charge.

Finally, note that Theorem 4.2 does not apply to discounted problems. To apply to discounted problems, we require a modified A3 that states that both functions  $d(0, u) + \lambda d(u, -u)$  and  $-\lambda d(0, u) - d(u, -u)$  are convex.

**Corollary 4.5.** Assume that the battery degradation function satisfies properties A1, A2, and modified A3. Then there exists an optimal two-threshold charge policy for the discounted infinite horizon problem with time-independent thresholds  $(c(\theta), C(\theta))$ .

The proof of Corollary 4.5 follows the same steps as the proof of Theorem 4.2 and is provided in the appendix. The stationarity follows using the standard argument for the existence of an optimal stationary policy, e.g. Theorem 6.2.7 in [Puterman, 2005].

The modified condition A3 is however more difficult to satisfy and verify than the original A3. However, it may be sufficient to satisfy either one of these properties approximately in order for a threshold policy be close to optimal. This analysis is, however, beyond the scope of the present paper.

## 5 OPTIMIZATION ALGORITHM

So far we described the structure of the optimal policy. It is important to also develop an algorithm that can take advantage of this structure and efficiently compute the optimal policy. In this section, we describe such an algorithm and prove its optimality. We focus on the infinite horizon problem which is more relevant in practice.

A naive approach to optimizing threshold policies is to iteratively evaluate a given set of thresholds by simulation and then optimize the threshold values. This class of methods is known as simulation-optimization or policy search [Carson and Maria, 1997]. Simply searching over all sets of thresholds is intractable because the number of threshold values that need to be computed is  $2^{|\Theta|}$ . As we show below this search can be decomposed by the states of the price process  $\theta$  leading to Algorithm 1. The algorithm optimizes each pair of thresholds independently for each state of the price process.

The evaluation function  $\tilde{f}(c_{\theta_1}, C_{\theta_1}, \dots, c_{\theta_k}, C_{\theta_k}, \dots, c_{\theta_n}, C_{\theta_n})$  is computed by simulating the execution. Using common random numbers when optimizing a value by simulation in this setting can significantly reduce sample variance and speed up the algorithm [Glasserman and Yao, 1992]. To model the discount factor, we assume a termination probability of  $1 - \lambda$  in every step. The function  $\tilde{f}$  is computed as a sample average.

In the remainder of the section, assume that the function  $\tilde{f}$  can be evaluated precisely. The result readily generalizes to

---

**Algorithm 1: Threshold Optimization by Simulation**


---

```

// Initialize thresholds
1  $(c_\theta, C_\theta) \leftarrow (0, 1) \quad \forall \theta \in \Theta$ ;
// Initialize step counter
2  $k \leftarrow 1$ ;
3  $g_0 \leftarrow \inf, g_{-1} \leftarrow \inf$ ;
4 while  $g_{k-1} < g_{k-2} + \epsilon$  do
5   for  $\theta \in \Theta$  do
6     // Optimize thresholds for  $\theta$ 
7      $c_\theta, C_\theta \leftarrow \arg \min_{\bar{c}_\theta, \bar{C}_\theta} \tilde{f}(\dots, \bar{c}_\theta, \bar{C}_\theta, \dots)$ ;
8     //  $\tilde{f}$  is sampled mean return
9      $g_k \leftarrow \tilde{f}(c_{\theta_1}, C_{\theta_1}, \dots, c_{\theta_n}, C_{\theta_n})$ ;
10     $k \leftarrow k + 1$ ;
// Return computed thresholds
9 return  $\{(\theta, c_\theta, C_\theta) : \theta \in \Theta\}$ 

```

---

the sampled setting by considering appropriate Hoeffding or Bernstein concentration inequalities.

**Proposition 5.1.** *Consider an energy arbitrage problem that satisfies the properties A1, A2, and modified A3 sufficient for the optimality of a threshold policy. Then Algorithm 1 converges to the optimal solution in a finite number of iterations.*

*Proof.* We argue that Algorithm 1 corresponds to a variant of the simplex algorithm implementation on the dual MDP formulation (e.g., [Puterman, 2005]). First, note that using the same argument as in the proof of Theorem 4.2 we can show that the optimal solution to the minimization in Algorithm 1 is also a threshold policy. Therefore, the algorithm corresponds to a coordinate descent on the linear program formulation of the MDP. The result then follows from the finite number of coordinate blocks.  $\square$

## 6 NUMERICAL RESULTS

This section numerically evaluates Algorithm 1 in an idealized, but realistic, model of daily energy price evolution and a degradable Li-ion battery. First, we analyze the properties of the computed solution and study the impact of the battery degradation on the quality of the computed policy. Then, we compare the solution based on a threshold policy to directly solving a discretized version of the problem.

The experimental setting assumes that energy is traded daily in a large exchange market that is not influenced by the trading policy. We compute policies for a discount factor of 0.9999 and report results of simulations of energy arbitrage throughout 5 years (1825 days).

Our energy prices are based on data from the Intercontinental Exchange (IEC) [IEC, 2015] for New England for years 2001 through 2013. The price per MW h in this period ranges between \$24 and \$312. Numerous papers have focused on building predictive models of energy

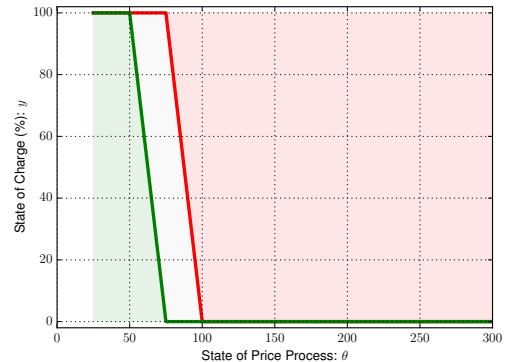


Figure 8: Optimal threshold policy  $\pi_{\text{non}}$  which does not consider battery degradation. The price state  $\theta$  represents the average price of energy in \$/MW h.

prices, most auto-regressive or latent [Mateo et al., 2005, Aggarwal et al., 2009]. Since the prediction problem is not the main focus of this work, we simply use a Markov model with quantized price data in \$25 intervals. The model is described in detail in Appendix A.1. More accurate models of the energy price, such as ones that include seasonal effects, may lead to significantly greater returns. In addition, to model transmission and battery inefficiencies, we use  $p^i$  that is 5% higher and  $p^o$  that is 5% lower than the spot market price.

The battery degradation process is based on a generic behavior of a Li-ion battery depicted in Fig. 2. The actual degradation will depend on the specific construction of the particular battery [Ramadesigan et al., 2012]. We assume a battery of size 1 MW h; using a larger battery would simply linearly scale the results. We assume a low price of Li-ion batteries at about \$20 per kW h, which translates to a degradation cost of  $c^d = 20000$ . While the current price of Li-ion batteries is considerably higher, it is expected to decrease in the future.

Policies are computed using 10 iterations of Algorithm 1. We first compute a policy  $\pi_{\text{non}}$  that ignores the effects of battery degradation. This is the approach taken by some previous relevant work [Harsha and Dahleh, 2011, Van de Ven et al., 2011]. This policy is depicted in Fig. 9. Second, we compute the policy that considers the degradation  $\pi_{\text{deg}}$  and show it in Fig. 9.

Both policies  $\pi_{\text{non}}$  and  $\pi_{\text{deg}}$  charge the battery to a relatively high level when the energy price is low and discharge it when the energy price is high. However, note that  $\pi_{\text{deg}}$  charges the battery to a lower maximal level, and also is more conservative in discharging the battery completely, unless the price of energy is especially high. This behavior decreases some potential trading revenues but minimizes battery degradation.

Fig. 10 compares the capacity loss of the two policies as a function of the trading day averaged over 10 runs. It is no-

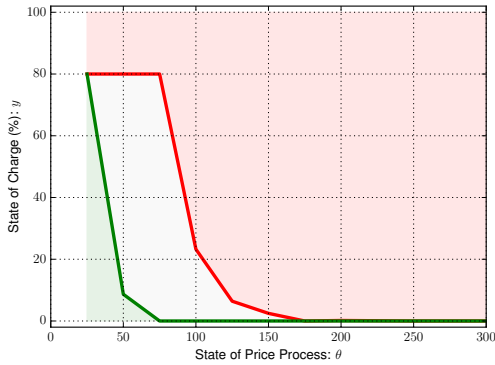


Figure 9: Optimal threshold policy  $\pi_{deg}$  which considers battery degradation. The price state  $\theta$  represents the average price of energy in \$/MWh.

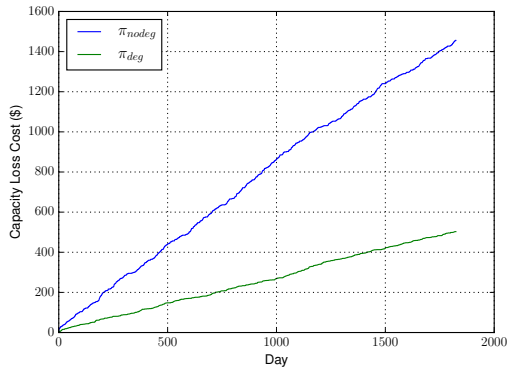


Figure 10: Cost of the capacity loss as a function of the trading day. The final capacity loss corresponds to about 10% of the initial capacity for  $\pi_{non}$ .

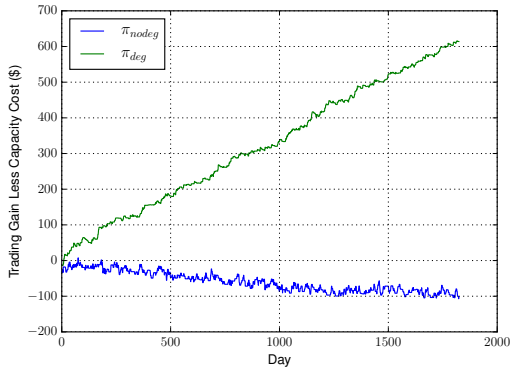


Figure 11: Cost of the capacity loss as a function of the trading day. The final capacity loss corresponds to about 10% of the initial capacity for  $\pi_{non}$ .

ticeable that the policy that does not consider ends up leading to battery degradation that is more than double of the policy that is optimized for battery loss. Fig. 11 shows the cumulative gains from the arbitrage less the cost of the lost battery capacity. Note that the policy that does not consider battery degradation ends up with negative returns. That is the cost of the lost capacity is greater than the profits earned from trading.

One note on practicality of using Li-ion batteries for energy storage is in order. Although we assumed a very low cost of Li-ion batteries, it does not appear that the return that we obtained is sufficient to offset the capital invested in the battery. However, when the capacity is already available for a different purpose, such as with an plug-in electric vehicle, energy arbitrage using the battery may be viable [Yudovina and Michailidis, 2014, Peterson et al., 2010].

## 7 CONCLUSION

We described sufficient conditions that guarantee the existence of an optimal threshold policy for energy arbitrage with a degradable battery storage. Threshold policies in this setting are very appealing for several reasons. They are relatively easy to compute and are simple to analyze, interpret, and implement. Our experimental results indicate that it is necessary to consider battery degradation in realistic scenarios since the battery cost is significant in comparison with energy prices. In addition, even when the degradation function does not satisfy the threshold policy assumptions, the proposed algorithm can compute very good solutions that in fact outperform a dynamic programming solution of the discretized problem.

Future work should characterize the structure of policies for problems that violate A3. In addition, the threshold policy property could be combine with a dynamic programming approach instead of simulation optimization. Such approach may lead to more efficient algorithms in terms of computational and sampling complexities.

### Acknowledgments

We thank Pavithra Harsha and Peter M. Van de Ven for extensive discussions of the results and for suggesting this research topic. We also thank the anonymous reviewers whose comments helped to improve the paper significantly.

### References

[IEC, 2015] (2015). Intercontinental Exchange: <http://www.eia.gov/electricity/wholesale>.

[Aggarwal et al., 2009] Aggarwal, S. K., Saini, L. M., and Kumar, A. (2009). Electricity price forecasting in deregulated markets: A review and evaluation. *International Journal of Electrical Power and Energy Systems*, 31(1):13–22.

[Aurbach, 2000] Aurbach, D. (2000). Review of selected electrodesolution interactions which determine the performance of

- Li and Li ion batteries. *Journal of Power Sources*, 89(2):206–218.
- [Bashash et al., 2011] Bashash, S., Moura, S. J., Forman, J. C., and Fathy, H. K. (2011). Plug-in hybrid electric vehicle charge pattern optimization for energy cost and battery longevity. *Journal of Power Sources*, 196(1):541–549.
- [Boyd and Vandenberghe, 2004] Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press, Cambridge.
- [Carson and Maria, 1997] Carson, Y. and Maria, a. (1997). Simulation optimization: methods and applications. *Proceedings of the 29th conference on Winter simulation*, pages 118–126.
- [Ermon et al., 2011] Ermon, S., Conrad, J., Gomes, C. P., and Selman, B. (2011). Risk-sensitive policies for sustainable renewable resource allocation. In *International Joint Conference on Artificial Intelligence*, pages 1942–1948.
- [Ermon et al., 2013] Ermon, S., Xue, Y., Gomes, C., and Selman, B. (2013). Learning policies for battery usage optimization in electric vehicles. *Machine learning*, 92(1):177–194.
- [Glasserman and Yao, 1992] Glasserman, P. and Yao, D. D. (1992). Some guidelines and guarantees for common random numbers. *Management Science*, 38(6):884–908.
- [Gomes, 2009] Gomes, C. P. (2009). Computational sustainability: Computational methods for a sustainable environment, economy, and society. *The Bridge*, 39(4):5–13.
- [Harsha and Dahleh, 2011] Harsha, P. and Dahleh, M. (2011). Optimal sizing of energy storage for efficient integration of renewable energy. In *IEEE Conference on Decision and Control and European Control Conference*, pages 5813–5819.
- [Hoke et al., 2011] Hoke, A., Brissette, A., Maksimovic, D., Pratt, A., and Smith, K. (2011). Electric vehicle charge optimization including effects of lithium-ion battery degradation. *2011 IEEE Vehicle Power and Propulsion Conference*, pages 1–8.
- [Lifshitz and Weiss, 2014] Lifshitz, D. and Weiss, G. (2014). Optimal Control of a Capacitor-Type Energy Storage System. *IEEE Transactions on Automatic Control*, 9286(1):1–6.
- [Mateo et al., 2005] Mateo, A., Muñoz, A., and García-González, J. (2005). Modeling and Forecasting Electricity Prices with Input/Output Hidden Markov Models. *IEEE Transactions on Power Systems*, 20(1):13–24.
- [Moura et al., 2011] Moura, S. J., Fathy, H. K., and Callaway, D. S. (2011). A stochastic optimal control approach for power management in plug-in hybrid electric vehicles. *IEEE Transactions on Control Systems Technology*, pages 1–11.
- [Nadarajah, 2014] Nadarajah, S. (2014). *Approximate Dynamic Programming for Commodity and Energy Merchant Operations*. PhD thesis, Carnegie Mellon.
- [Peterson et al., 2010] Peterson, S. B., Apt, J., and Whitacre, J. F. (2010). Lithium-ion battery cell degradation resulting from realistic vehicle and vehicle-to-grid utilization. *Journal of Power Sources*, 195:2385–2392.
- [Petrik and Zilberstein, 2011] Petrik, M. and Zilberstein, S. (2011). Linear dynamic programs for resource management. In *Conference on Artificial Intelligence (AAAI)*.
- [Porteus, 2002] Porteus, E. L. (2002). *Foundations of Stochastic Inventory Theory*. Stanford Business Books.
- [Puterman, 2005] Puterman, M. L. (2005). *Markov decision processes: Discrete stochastic dynamic programming*. John Wiley & Sons, Inc.
- [Ramadass et al., 2003] Ramadass, P., Haran, B., White, R., and Popov, B. N. (2003). Mathematical modeling of the capacity fade of Li-ion cells. *Journal of Power Sources*, 123(2):230–240.
- [Ramadesigan et al., 2012] Ramadesigan, V., Northrop, P. W. C., De, S., Santhanagopalan, S., Braatz, R. D., and Subramanian, V. R. (2012). Modeling and Simulation of Lithium-Ion Batteries from a Systems Engineering Perspective. *Journal of The Electrochemical Society*, 159(3):31–44.
- [Serrao et al., 2005] Serrao, L., Chehab, Z., Guezennec, Y., and Rizzoni, G. (2005). An Aging Model of Ni-MH Batteries for Hybrid Electric Vehicles. *2005 IEEE Vehicle Power and Propulsion Conference*, pages 78–85.
- [Sion, 1958] Sion, M. (1958). On general minimax theorems. *Pacific Journal of Mathematics*, 8(4):171–176.
- [Van de Ven et al., 2011] Van de Ven, P., Hegde, N., Massoulié, L., and Salonidis, T. (2011). Optimal Control of Residential Energy Storage Under Price Fluctuations. *ENERGY 2011, The First International Conference on Smart Grids, Green Communications and IT Energy-aware Technologies*, pages 159–162.
- [Walawalkar et al., 2007] Walawalkar, R., Apt, J., and Mancini, R. (2007). Economics of electric energy storage for energy arbitrage and regulation in New York. *Energy Policy*, 35(4):2558–2568.
- [Yudovina and Michailidis, 2014] Yudovina, E. and Michailidis, G. (2014). Socially Optimal Charging Strategies for Electric Vehicles. *IEEE Transactions on Automatic Control*, 9286(c):1–6.

---

# Mesochronal Structure Learning

---

**Sergey Plis**

Mind Research Network &  
University of New Mexico  
Albuquerque, NM 87106

**David Danks**

Department of Philosophy  
Carnegie Mellon University  
Pittsburgh, PA 15213

**Jianyu Yang**

Mind Research Network &  
University of New Mexico  
Albuquerque, NM 87106

## Abstract

Standard time series structure learning algorithms assume that the measurement timescale is approximately the same as the timescale of the underlying (causal) system. In many scientific contexts, however, this assumption is violated: the measurement timescale can be substantially slower than the system timescale (so intermediate time series datapoints will be missing). This assumption violation can lead to significant learning errors. In this paper, we provide a novel learning algorithm to extract system-timescale structure from measurement data that undersample the underlying system. We employ multiple algorithmic optimizations that exploit the problem structure in order to achieve computational tractability. The resulting algorithm is highly reliable at extracting system-timescale structure from undersampled data.

## 1 INTRODUCTION

In many domains, measurement speed can be significantly slower than the causal or communication speeds in the underlying system. For example, fMRI experiments typically measure brain activity roughly every two seconds, but the causal and communication connections between neuronal layers operate much faster [8]. Similar observations can be made about systems in ecology, climatology, economics, genomics and proteomics, and cognitive science. Moreover, a discrepancy between the measurement timescale  $\tau_M$  and the system timescale  $\tau_S$  can make a difference: an apparent  $A \rightarrow B$  connection at  $\tau_M$  can be consistent with any possible connection at  $\tau_S$ :  $A \rightarrow B$ ,  $A \leftarrow B$ , or no connection at all. Thus, it is critical that we not simply restrict our attention to learning connections at  $\tau_M$ .

In this paper, we address the problem of learning the causal structure at  $\tau_S$  from measurements taken at a slower sam-

pling rate, also called “undersampled” data.<sup>1</sup> We focus on cases in which the underlying system structure can be represented as a directed graphical model (without simultaneous influence). There has been very little prior work on the problem of structure learning from undersampled time series data, though there have been important prior explorations of learning when the measurement and system timescales diverge, or when causal influences operate on multiple timescales [3, 5, 9]. There are multiple algorithms for learning graphical structure from time series data [6, 7, 11, 14, 15], but they all assume that  $\tau_M$  is at least as fast as  $\tau_S$ . Undersampling was explicitly addressed in [2], but they focused on the “forward” problem of undersampling: given a structure at  $\tau_S$ , what structure will be realized at  $\tau_M$ ? That paper provided some preliminary theorems (used below) to characterize the backward problem, but not a usable algorithm for actually learning structure at  $\tau_S$  from measurements at  $\tau_M$ . In this paper, we introduce such an algorithm: the *Mesochronal Structure Learning* (MSL) algorithm (from Greek μέσω (*mésō*) for “through” and χρόνος (*chrónos*) for “time”) (Section 3); and show that it can often learn significant  $\tau_S$  structure from  $\tau_M$  data (Section 4). First, however, we provide a precise statement of the problem.

## 2 REPRESENTATION AND FORMALISM

We use a compressed graph representation of the underlying system structure.<sup>2</sup> We assume that the system is first-order Markov,<sup>3</sup> and so temporal information can be encoded directly in the graphical edges. This assumption also implies a form of “causal sufficiency”: specifically,

---

<sup>1</sup>Measurements taken at a faster sampling rate pose a computational challenge, but not a distinctive theoretical problem.

<sup>2</sup>This framework is mathematically equivalent to dynamic Bayesian networks [4, 12], so all results could instead be expressed using DBNs [2]. However, compressed graphs provide significant computational advantages for this particular problem domain.

<sup>3</sup>That is, the system-state at  $t$  is independent of all system-states at  $t - n$  for  $n > 1$ , conditional on the system-state at  $t - 1$ .



there cannot be unobserved variables such that nodes in the current timestep (at the causal timescale) are conditionally associated once the variable values at the previous timestep are known. Let  $\mathcal{G}$  be a directed graphical model over variables  $\mathbf{V}$  such that  $V_i \rightarrow V_j$  means  $V_i^{t-1} \rightarrow V_j^t$ , where superscripts denote (relative) time index. We exclude contemporaneous connections because  $\tau_S$  can be arbitrarily fast.  $\mathcal{G}$  can be cyclic, including self-loops, but the underlying system structure will be acyclic when “unrolled” through time. Let  $P(2\mathbf{V})$  be a joint probability distribution over  $\mathbf{V}^t$  and  $\mathbf{V}^{t-1}$ . We connect  $\mathcal{G}$  and  $P(2\mathbf{V})$  through standard assumptions, though adjusted for this setting. Specifically, let  $\text{pa}(V_i)$  denote the parents of  $V_i$  in  $\mathcal{G}$ . The Markov assumption requires:  $V_i^t$  is independent of  $[\mathbf{V}^t \setminus V_i^t] \cup [\mathbf{V}^{t-1} \setminus \text{pa}(V_i)^{t-1}]$  conditional on  $\text{pa}(V_i)^{t-1}$ . The Faithfulness assumption requires that these be the only independencies involving some  $V_i^t$ .

Let  $\{t^0, t^1, \dots, t^k, \dots\}$  denote the timesteps at the system timescale. We say that the system is *sampled at rate  $u$*  when the measured timesteps are  $\{t^0, t^u, \dots, t^{ku}, \dots\}$ . The system timescale is thus “sampled at rate 1.” We focus on cases of undersampling; that is, when  $u > 1$ . Undersampling implies failure to observe intermediate steps on paths between variables, and so the measurement timescale graph  $\mathcal{G}^u$  can be derived from the causal timescale graph  $\mathcal{G}^1$ . More precisely,  $V_i \rightarrow V_j$  in  $\mathcal{G}^u$  iff there is a path of length  $u$  from  $V_i$  to  $V_j$  in  $\mathcal{G}^1$ . Undersampling can also introduce bidirected edges that represent unobserved common causes of variables at time  $t$ . For example, if  $V_i \leftarrow V_c \rightarrow V_j$  in  $\mathcal{G}^1$ , then for all  $u > 1$ ,  $\mathcal{G}^u$  will contain  $V_i \leftrightarrow V_j$  since the unmeasured  $V_c^{t-1}$  is a parent of both  $V_i^t$  and  $V_j^t$ . If the true system structure  $\mathcal{G}^1$  and the sampling rate  $u$  are known, then there are efficient algorithms for computing the resulting (expected) measurement timescale structure  $\mathcal{G}^u$  [2].

The general problem of inferring  $\mathcal{G}^1$  from data sampled at unknown rate  $u$  is computationally intractable at the current time, and so we principally focus on the special case in which  $u = 2$  (though Section 4 shows how to generalize our algorithm to  $u > 2$ ). That is, what can be learned about  $\mathcal{G}^1$  if the input data is a time series in which every other timestep is unobserved? It is straightforward to see that  $\mathcal{G}^2$  can be quite different from  $\mathcal{G}^1$ ; for example, if  $\mathcal{G}^1$  is a directed cycle over three variables (e.g.,  $X \rightarrow Y \rightarrow Z \rightarrow X$ ), then that cycle will have the reverse direction in  $\mathcal{G}^2$ . At the same time,  $\mathcal{G}^2$  and  $\mathcal{G}^1$  cannot be arbitrarily different; for example, if  $V_i \rightarrow V_i$  in  $\mathcal{G}^1$  (i.e.,  $V_i$  has a self-loop), then  $V_i \rightarrow V_i$  in  $\mathcal{G}^2$ . We now provide a multi-step algorithm for recovering as much information as possible.

### 3 MSL ALGORITHM

There are a number of previously identified structural invariants of  $\mathcal{G}^1$  that hold across sampling rates [2], but many of them provide only a coarse characterization of the struc-

ture of  $\mathcal{G}^1$ . We thus must search in a more direct fashion for the  $\mathcal{G}^1$  that could have produced  $\mathcal{G}^2$ . The Mesochronal Structure Learning (MSL) algorithm has two distinct steps. First, one learns  $\mathcal{G}^2$  from data, expert knowledge, or a combination of the two (Section 3.1). There are many different algorithms for learning causal structure at the measurement timescale (i.e.,  $\mathcal{G}^2$ ), and so we focus on the second step: infer the set of  $\mathcal{G}^1$  that could possibly have produced (given undersampling) the learned  $\mathcal{G}^2$  (Section 3.2). The  $\mathcal{G}^2 \rightarrow \mathcal{G}^1$  mapping is one-to-many, and so the MSL algorithm outputs an equivalence class (possibly a singleton) of possible  $\mathcal{G}^1$ . The MSL algorithm is based on a conceptually simple inferential move, but requires significant algorithmic (Section 3.3) and practical (Section 3.4) optimizations in order to be computationally tractable.

#### 3.1 LEARNING $\mathcal{G}^2$

There are many different algorithms for learning the structure of  $\mathcal{G}^2$  from time series data [6, 7, 11, 14, 15], as the measurement and structure timescales are the same. One can also modify structure learning algorithms designed for i.i.d. data (e.g., the well-known PC or GES algorithms [1, 13]) for the special case of time series data in which the causal direction can be inferred from temporal information. We will mostly treat these algorithms as “black boxes” that simply provide an estimated  $\mathcal{G}^2$  for input to the second stage. We cannot completely abstract away from details of those algorithms, however, since errors learning  $\mathcal{G}^2$  structure can result in errors by the overall MSL algorithm. We return to this issue in Section 4, but focus for now on the algorithmically novel aspect of inferring causal timescale structure from estimated measurement timescale structure.

#### 3.2 FROM $\mathcal{G}^2$ TO $\mathcal{G}^1$

Given a known  $\mathcal{G}^1$  and undersample rate  $u$ , [2] provides an efficient method for computing  $\mathcal{G}^u$ . Thus, for an estimated  $\mathcal{H}^2$ , there is an obvious brute-force approach: for all  $\mathcal{G}^1$ , compute the corresponding  $\mathcal{G}^2$  and check if it equals the estimated  $\mathcal{H}^2$ . The problem with this approach is equally obvious: it must survey every possible  $\mathcal{G}^1$ , of which there are  $2^{n^2}$  many. This brute-force strategy could potentially work for 3-, 4-, or even 5-node graphs, but rapidly becomes computationally completely infeasible. We thus pursue a different strategy.

We focus throughout on the case of a single Strongly Connected Component (SCC): a maximal variable set  $\mathbf{S}$  such that there is a path from every  $X \in \mathbf{S}$  to every  $Y \in \mathbf{S}$ . All systems with feedback are composed of SCCs, and so they are the most scientifically interesting systems when working with time series data. When a very weak additional condition holds,<sup>4</sup> then SCC membership is invariant under

<sup>4</sup>Every SCC  $\mathbf{S}$  can be uniquely expressed as the union of a set

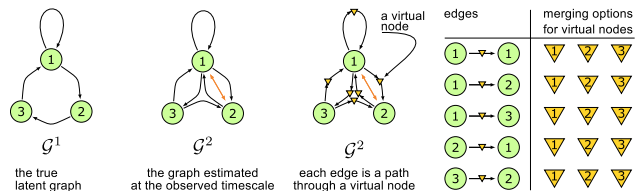


Figure 1: A 3-node SCC at undersampling rates 1 and 2, as well as its virtual nodes and their merging options.

undersampling (Corollary 7 in [2]). Thus, we can use  $\mathcal{G}^2$  structure to reliably identify SCC membership in  $\mathcal{G}^1$ , and then do focused search over each SCC separately.

Theorems 4 and 5 in [2] show that, when  $u$  gets very large, such an SCC becomes a *super-clique*: for every pair of nodes  $A, B$  (possibly  $A = B$ ), we have  $A \rightarrow B$ ,  $A \leftarrow B$ , and  $A \leftrightarrow B$ . (The last two do not apply when  $A = B$ .) That is, a super-clique is a maximally dense graph over the SCC. Moreover, these super-cliques are the worst-case for a “backwards” learning algorithm, as a huge number of SCCs imply a super-clique under (significant) undersampling. Thankfully, super-cliques rarely result for smaller undersample rates; typically, more can be learned at  $u = 2$ .

Given an estimated  $\mathcal{H}^2$  that is an SCC, every directed edge corresponds to a path of length 2 in  $\mathcal{G}^1$ . More generally, if we have estimated  $\mathcal{H}^u$  for a known  $u$ , then each edge must correspond to a path of length  $u$  in  $\mathcal{G}^1$ . Thus, we can add  $u - 1$  “virtual” nodes within each edge in  $\mathcal{H}^u$ , where each virtual node refers to some unknown, but actual, node in  $\mathbf{V}$ . The virtual-to-actual node mapping can clearly be many-to-one, as  $u$  can be significantly larger than the size of  $\mathbf{V}$ . This virtual node representation is shown in Figure 1.

The basic structure of this stage of the MSL algorithm is: (1) “identify” each virtual node with an actual node, thereby yielding a candidate  $\mathcal{G}^1$ ; and then (2) check if that candidate actually implies  $\mathcal{H}^u$ . As noted above, there is a computationally efficient algorithm for step (2); the computational challenge is efficiently considering the relevant possible identifications. We focus in the remainder of this section on the case of  $u = 2$  as that is sufficient to reveal significant complexities. The overall algorithm-schema is importantly not limited to that case, however, and we provide a “proof-of-concept” for  $u = 3$  in Section 4.

For  $e$  edges in  $\mathcal{H}^2$ , there are  $n^e$  possible node identifications,<sup>5</sup> each of which results in a candidate  $\mathcal{G}^1$ . Moving directly to complete identifications can require examining an intractable number of  $\mathcal{G}^1$  (e.g., if  $n > 30$  and  $e > 100$ , as below). We thus instead sequentially identify virtual nodes, coupled with a (local) stopping rule based on the concept

of simple loops  $\mathcal{L}_S$ . Let  $\text{gcd}(\mathcal{L}_S)$  be the greatest common divisor of the lengths of those simple loops. The additional condition is that  $\text{gcd}(\mathcal{L}_S) = 1$ .

<sup>5</sup>In general, there are  $n^{e(u-1)}$  possible identifications.

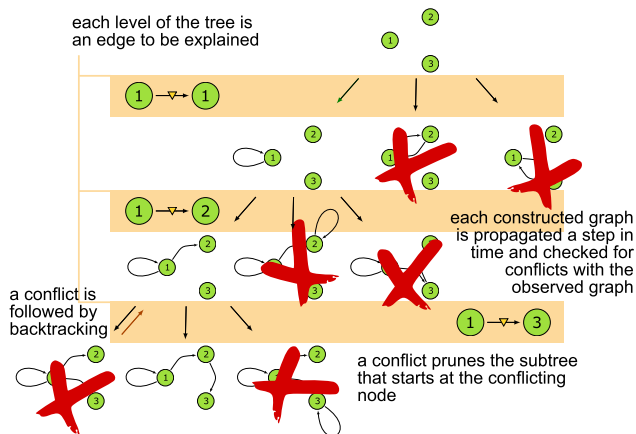


Figure 2: The search tree for 3-node SCC of Figure 1

of a *conflict*, and a lemma (with corollary):<sup>6</sup>

**conflict**  $\mathcal{G}^u$  contains one or more edges that are not in  $\mathcal{H}^u$ .

**Lemma 3.1. Conflict persistence:** *If a virtual node identification results in a conflict, then no further node identifications will eliminate that conflict.*

**Corollary 3.2.** *If  $\mathcal{G}^1$  conflicts with  $\mathcal{H}^u$ , then every super-graph of  $\mathcal{G}^1$  conflicts with  $\mathcal{H}^u$ .*

Thus, if any partial virtual node identification results in a  $\mathcal{G}^1$  whose  $\mathcal{G}^u$  contains an edge not found in  $\mathcal{H}^u$ , then we need not consider any further identifications that build off of that base. This naturally suggests a backtracking search on a search tree through the possible node identifications, as shown in Figure 2. More precisely, the basic MSL algorithm is:

1. Let  $\mathcal{G}$  be the empty graph, and  $\{E_1, \dots, E_e\}$  be an arbitrary ordering of edges in  $\mathcal{H}^u$  (Table in Figure 1)
2. In a depth-first manner over the edges, consider each possible node identification for the virtual nodes added to  $E_i$  and add the corresponding edges to  $\mathcal{G}$
3. Check whether a conflict is found after adding the edges arising from virtual node identification for  $E_i$
4. If a conflict is found, then prune that search tree branch, backtrack by removing the  $E_i$  identification, and try the next possible node identifications for  $E_i$ .

The process is illustrated in Figure 2 for the graph in Figure 1. In the worst case, this algorithm obviously requires checking as many  $\mathcal{G}^1$  as if we simply surveyed all possible simultaneous node identifications. In practice, however, the proactive pruning of branches in the search tree can lead to considerable speed-ups, especially in cases in which  $e$  is relatively large.

This algorithm is correct but not yet complete, as  $\mathcal{G}^1$  can contain edges that do not have manifest in any way in  $\mathcal{G}^2$ . For example, if  $\mathcal{G}^1$  is  $A \rightarrow B$ , then  $\mathcal{G}^2$  is simply the empty

<sup>6</sup>All proofs are provided in Supplementary Materials.

graph over  $A, B$ . In that case, there are no virtual nodes to identify, so the algorithm would correctly but incompletely return the empty graph as the only  $\mathcal{G}^1$  possibility. More generally, especially for relatively dense  $\mathcal{H}^2$ , the algorithm finds a suitable  $\mathcal{G}^1$  prior to reaching the full depth of the search tree (i.e., without identifying all virtual nodes). One response would be to simply force the algorithm to fully traverse the tree, but this can be quite expensive when the branching factor is high (i.e., for dense  $\mathcal{H}^2$ ). Instead, we pursue a different strategy.

If the algorithm finds a suitable  $\mathcal{G}^1$  before reaching a leaf of the search tree, then we know that every graph below it in the tree will be a supergraph of that  $\mathcal{G}^1$  (since virtual node identifications can only add edges, not remove them). Thus, we only need to find all supergraphs of that  $\mathcal{G}^1$  whose  $\mathcal{G}^2 = \mathcal{H}^2$ . That search is greatly aided by Corollary 3.2.

The supergraph construction step first tries to separately add each of the  $n^2$  possible directed edges that are not yet in  $\mathcal{G}^1$ . Each resulting graph that equals  $\mathcal{H}^2$  is added to the output equivalence class. The step then adds, in a depth-first manner, each edge that did not yield a conflict to the other new graphs, and backtracks whenever an edge addition creates a conflict.<sup>7</sup> This step is extremely fast in practice for graphs of reasonable sparsity despite its worst-case factorial behavior. If no edges create a conflict—for example, when  $\mathcal{H}^2$  is a super-clique—then the running time is indeed  $\Theta(n!)$ . In that particular case, however, the equivalence class has been analytically determined to be any size- $n$  SCC with  $\text{gcd}=1$  (see fn. 4) [2, Theorem 4], and so the present algorithm is actually unnecessary.

The full MSL algorithm (including the supergraph step) has the following desirable property:

**Lemma 3.3.** *The MSL algorithm is correct and complete: given  $\mathcal{H}^2$ , it finds all and only  $\mathcal{G}^1$  such that  $\mathcal{G}^2 = \mathcal{H}^2$ .*

Unfortunately, preliminary experiments demonstrated that the algorithm can be very slow (see Figure 4 for a highlight of the problem) and take days even for smaller ( $n = 10$ ) graphs. The order in which virtual nodes are identified can make a significant difference in runtime speed, but even improving those orders is insufficient to yield an algorithm that is usable on large graphs. Instead, we must exploit additional constraints and optimizations.

### 3.3 USING GRAPHICAL CONSTRAINTS

The key intuition underlying the constraints in this section is that some virtual node identifications can be excluded without ever actually constructing-and-testing the corresponding  $\mathcal{G}^1$ . For example, suppose  $A \rightarrow B \rightarrow C$  in  $\mathcal{H}^2$ . In this case,  $\mathcal{G}^1$  must contain, for some  $X, Y$ :  $A \rightarrow X \rightarrow B \rightarrow Y \rightarrow C$ . There is thus a length-2 path from  $X$  to  $Y$  in  $\mathcal{G}^1$ , and so  $\mathcal{G}^2$  will contain  $X \rightarrow Y$ .

<sup>7</sup>See pseudocode in the Supplementary Material.

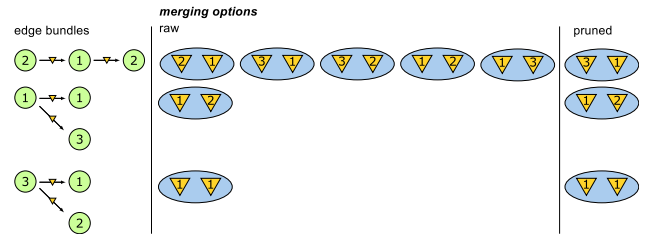


Figure 3: Edge-pairs for 3-node SCC of Figure 1 and merging options for their virtual nodes: all possible options (raw) and the one that remain after constructing the pairwise data structure (pruned).

Hence, we only need to consider virtual node identifications for  $A \rightarrow B$  and  $B \rightarrow C$  in which the two identifications correspond to nodes with a directed edge between them in  $\mathcal{H}^2$ . More generally, virtual node identifications can analytically constrain one another in ways that can be exploited in this algorithm.

Recall that the complexity of the MSL algorithm for  $u = 2$  is approximately  $n^e$ , where  $e$  is the number of edges in  $\mathcal{H}^2$ . By identifying pairs of virtual nodes (that analytically constrain one another), we can potentially achieve a large reduction in the exponent in practice, since we will have to consider many fewer branches.

Two different types of structures in  $\mathcal{H}^2$  guide the pairwise identifications. First, consider all *forks* in  $\mathcal{H}^2$ : pairs of edges  $X \leftarrow H \rightarrow Y$ , where possibly  $X = H$  or  $Y = H$  (if there is a self-loop plus another edge). If the two virtual nodes refer to the same actual node, then  $X$  and  $Y$  will have a common cause in the previous (causal) timestep, and so there will be a bidirected edge between them.<sup>8</sup> Thus, if there is no bidirected edge between  $X$  and  $Y$  in  $\mathcal{H}^2$ , then the two virtual nodes cannot identify to the same node. Hence, we only need to consider  $n^2 - n$  possible identifications for that pair of virtual nodes.

The other relevant structure is the two-edge chain described at the start of this section, where the only pairwise virtual node identifications that are considered are those for which there is a corresponding edge in  $\mathcal{H}^2$ .

In practice, the algorithm converts some elements of the edge list  $\{E_1, \dots, E_e\}$  into edge-pairs by first selecting (without replacement) all forks in  $\mathcal{H}^2$ , then selecting all remaining two-edge directed paths. The remaining edges have the usual  $n$  possible virtual node identifications.

Figure 3 shows a search space for the graph from Figure 1, and demonstrates the computational advantage of considering pairwise identifications, as the number of possible identifications is significantly reduced.

<sup>8</sup>Note that the converse does not hold:  $X \leftrightarrow Y$  in  $\mathcal{H}^2$  does not imply that the virtual nodes correspond to the same actual node.

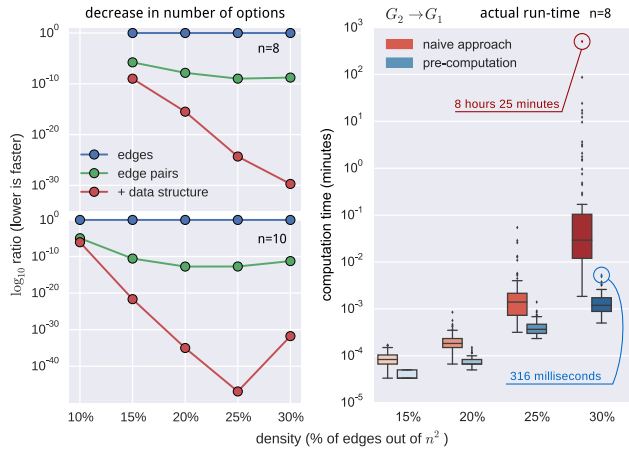


Figure 4: Comparison of the search-space size (left) and computation time (right) between the naive backtracking and our approaches.

In general, let  $\{m_1, \dots, m_l\}$  be the sets of virtual node identifications for each of the edges or edge-pairs derived from the preceding procedure. The computational complexity of using some edge-pairs is simply  $\prod_i \text{len}(m_i)$ , where  $\text{len}$  is the number of possible identifications for that particular edge or edge-pair. Thus, the computational advantage, expressed as a log-ratio, of using (some) edge-pairs is:  $\log r = \sum_i \log \text{len}(m_i) - e \log n$ . This advantage is plotted in Figure 4, which shows (on log-scale) the average log-ratio for 100 random 8- and 10-node SCCs. The practical advantage of edge-pairs is potentially even greater, as Figure 4 does not account for active pruning of the search tree.

### 3.4 PRECOMPUTATION AND OPTIMIZATIONS

The use of edge-pairs provides significant speed-up in the MSL algorithm (as seen in Figure 4), but its worst-case complexity still makes it difficult to use the algorithm for  $n > 10$ . Moreover, around 10% of the graphs took many days to compute, and did not exhibit any noticeable structural difference in  $\mathcal{H}^2$  that could be used to predict computation time.<sup>9</sup> The core problem is that much of the search tree eventually gets pruned, but structural features of  $\mathcal{H}^2$  do not support predictions about *which* parts will be pruned. If we can instead prune conflicting options prior to the traversal of the solution space, then we can potentially further reduce the search time.

The key algorithmic move is to expand the “analytic conflict checking” beyond just particular edge-pairs, to also pre-computing conflicts between multiple edge-pairs. In the extreme, there can be a (seemingly) possible virtual

<sup>9</sup>Even with random restarts to account for order differences in edge-pair selection, the algorithm still behaved similarly to the “only single edge” version.

node identification that conflicts with *every* possible identification for some other edge or edge-pair, in which case the first identification can simply be removed from consideration. Moreover, this precomputation is independent of the evaluation order of the virtual node identifications, since a conflict between identifications  $m_i$  and  $m_j$  does not depend on the particular values  $i$  and  $j$ . This pruning thus also reduces the need for random restarts to be robust against evaluation order effects.

The MSL algorithm was therefore expanded to include significant pre-computation to further prune the initial search tree. Specifically, for each pair of edge-pairs, the algorithm constructs a  $\mathcal{G}^1$  for every possible identification of both edge-pairs. If the resulting  $\mathcal{G}^1$  results in a conflict with  $\mathcal{H}^2$ , then we remove that pair of complex identifications from the search tree. This precomputation can considerably prune the search tree, perhaps even yielding (as in Figure 3) a search “tree” with only one branch. There is a cost because the resulting data structure can be complex: it contains not only the reduced set of (complex) identifications for each edge-pair, but also splits those options into subsets depending on the particular (complex) identification used in the previous level of the search tree. This more complex structure is required because some possible identifications will be incompatible with only a subset of the identifications at the previous level, but we want to avoid checking them during the algorithm flow (since we already checked in the precomputation).

Further algorithmic speed-ups can be achieved by intelligently ordering the virtual node identifications (i.e., the levels of the search tree). In particular, the MSL algorithm will run fastest when search tree levels with significant breadth—that is, virtual nodes or node-pairs for which there are many possible identifications—are pushed further down in the search tree. If this is done, then pruning operations will remove more branches. In addition, each search tree node is also a “conflict check” point, and this ordering of search tree levels minimizes the number of search tree nodes, *even if no branches are ever pruned*. The benefits of intelligent identification ordering can be substantial for deep trees with small numbers of possible identifications for most of the edge-pairs.

These two optimizations—precomputation and intelligent search tree ordering—yield substantial benefits. We again computed the potential reduction in computation for randomly generated graphs (using the equation from Section 3.3). The “+ data structure” line on the left-hand plot in Figure 4 shows that the more optimized approach can achieve over 40 orders of magnitude reduction in the number of conflict checks.

Each conflict check requires the algorithm to add edges to the constructed  $\mathcal{G}^1$ , compute  $\mathcal{G}^2$ , check for conflicts with  $\mathcal{H}^2$ , and then remove the just-added edges in case of a con-

flict. These steps are computationally expensive, and so we can achieve further performance gains if we can analytically determine whether a complex identification creates a conflict before starting these operations. These checks are not precomputed, but rather are performed in an online fashion based on the constraints in the following lemmas (where  $ch_{\mathcal{G}}(A)$  and  $pa_{\mathcal{G}}(A)$  denote the children and parents of  $A$  in  $\mathcal{G}$ , respectively):

**Lemma 3.4.** *A virtual node  $V$  in  $S \xrightarrow{V} E$  cannot be identified with node  $X$  if any of the following holds:*

1.  $\exists W \in ch_{\mathcal{G}^1}(S) \setminus X$  s.t.  $\nexists W \leftrightarrow X \in \mathcal{H}^2$
2.  $\exists W \in ch_{\mathcal{G}^1}(X) \setminus E$  s.t.  $\nexists W \leftrightarrow E \in \mathcal{H}^2$
3.  $\exists W \in ch_{\mathcal{G}^1}(X)$  s.t.  $\nexists S \rightarrow W \in \mathcal{H}^2$
4.  $\exists W \in ch_{\mathcal{G}^1}(E)$  s.t.  $\nexists X \rightarrow W \in \mathcal{H}^2$
5.  $\exists W \in pa_{\mathcal{G}^1}(S)$  s.t.  $\nexists W \rightarrow X \in \mathcal{H}^2$
6.  $\exists W \in pa_{\mathcal{G}^1}(X)$  s.t.  $\nexists W \rightarrow E \in \mathcal{H}^2$

**Lemma 3.5.** *A virtual node pair  $V_1, V_2$  for a fork  $E_1 \xleftarrow{V_1} S \xrightarrow{V_2} E_2$  cannot be identified with nodes  $X_1, X_2$  if any of the following holds:*

1.  $V_1$  in  $E_1 \xleftarrow{V_1} S$  cannot be identified with  $X_1$
2.  $V_2$  in  $S \xrightarrow{V_2} E_2$  cannot be identified with  $X_2$
3.  $V_1 \equiv E_2 \wedge V_2 \notin pa_{\mathcal{H}^2}(E_1)$
4.  $V_2 \equiv E_1 \wedge V_1 \notin pa_{\mathcal{H}^2}(E_2)$
5.  $S \equiv V_2 \wedge V_1 \neq V_2 \wedge V_1 \neq E_2$  and  $\nexists E_2 \leftrightarrow V_1 \in \mathcal{H}^2$
6.  $S \equiv V_1 \wedge (V_1 \equiv V_2 \vee V_2 \equiv E_2)$  and  $\nexists E_1 \leftrightarrow E_2 \in \mathcal{H}^2$
7.  $S \equiv V_1 \wedge (V_1 \equiv V_2 \vee V_2 \equiv E_2)$  and  $\nexists E_1 \leftrightarrow E_2 \in \mathcal{H}^2$
8.  $S \equiv V_2 \wedge (V_1 \equiv V_2 \vee V_1 \equiv E_1)$  and  $\nexists E_1 \leftrightarrow E_2 \in \mathcal{H}^2$
9.  $V_1 \equiv V_2$  and  $\nexists E_1 \leftrightarrow E_2 \in \mathcal{H}^2$

**Lemma 3.6.** *A virtual node pair  $V_1, V_2$  for two-edge sequence  $S \xrightarrow{V_1} M \xrightarrow{V_2} E$  cannot be identified with  $X_1, X_2$  if any of the following holds:*

1.  $V_1$  in  $S \xrightarrow{V_1} M$  cannot be merged to  $X_1$
2.  $V_2$  in  $M \xrightarrow{V_2} E$  cannot be merged to  $X_2$
3.  $V_1 \equiv V_2 \wedge (M \notin pa_{\mathcal{H}^2}(M) \vee V_1 \notin pa_{\mathcal{H}^2}(V_2) \vee S \notin pa_{\mathcal{H}^2}(E))$

As noted above, we apply these constraints in an online manner in order to prune branches of the search tree without having to add edges and check for conflicts. The resulting algorithm exhibits substantial reductions in runtime, enabling us to examine the MSL algorithm’s behavior for  $\mathcal{G}^1$  ranging up to 35 nodes, as shown in the simulation testing described in the next section.

## 4 TESTING AND VALIDATION

As we argued earlier, SCCs represent the scientifically most interesting situations, precisely because they are ones in which feedback loops present a challenging learning task. In addition, although connections between SCCs are

undoubtedly of interest, the formal results of [2] imply that we can reliably treat the SCCs relatively independently. We thus focus on single-SCC graphs in our synthetic data studies. Any SCC can provably be decomposed into a single simple loop with “ears” (i.e., sequences that branch off from, then return to, that simple loop) that build on top of one another. We thus use a simple ear decomposition skeleton to generate SCCs for our simulations.

**SCC generation procedure:** For  $n$  nodes, first generate a single simple loop that passes through all nodes. Without loss of generality, we can assume that this ring graph passes through the nodes in sequential order. There are  $n(n-1)$  possible edges that can be added to this ring graph, including self-loops for each node. We sample uniformly from those possible edges until the required density—i.e., the fraction of the  $n^2$  possible edges that are actual—is achieved.<sup>10</sup> We use overall density rather than average node degree to measure graph complexity because density is normalized by the number of possible edges, so we can (approximately) match graph complexity across different values of  $n$ .

We previously reported the theoretical maximum conflict checks for 8- and 10-node graphs for different versions of the MSL algorithm. We also report (in Figure 4) a comparison of the actual run times. We randomly generated 100 8-node graphs for each density in  $\{15\%, 20\%, 25\%, 30\%\}$  and ran both the naive approach (virtual node identification for each edge separately) and the precomputation approach that takes advantage of edge-pairs and pairwise constraints. The box and whisker plot shows the distribution of the individual run-times expressed in minutes. Not only does the median execution time for 8-node graphs improve by an order of magnitude, but the naive approach also generates considerably more outliers that take much longer to compute. The run-times for the hardest graphs (which provide an empirical estimate on the run-time upper bound) are five orders of magnitude longer for the naive approach.<sup>11</sup>

### 4.1 EQUIVALENCE CLASS SIZES

We earlier noted that, when  $\mathcal{H}^2$  is a super-clique (i.e., every possible edge between each pair of nodes), then there will typically be a large number of  $\mathcal{G}^1$  consistent with that super-clique. That is, the equivalence class will be quite large. One question is about the sizes of the equivalence classes when  $\mathcal{H}^2$  is *not* a super-clique. If the equivalence class is sufficiently small, then expert knowledge or further studies may be a tractable way to reach a unique solution.

To better explore the equivalence class sizes, we generated

<sup>10</sup>Note that the bare ring graph has a density of  $1/n$ .

<sup>11</sup>This may be less relevant in practice, except for the unlucky 10% of researchers who happen to deal with the hardest graphs. Nonetheless, this difference is substantial, and shows the importance of the algorithmic optimizations.

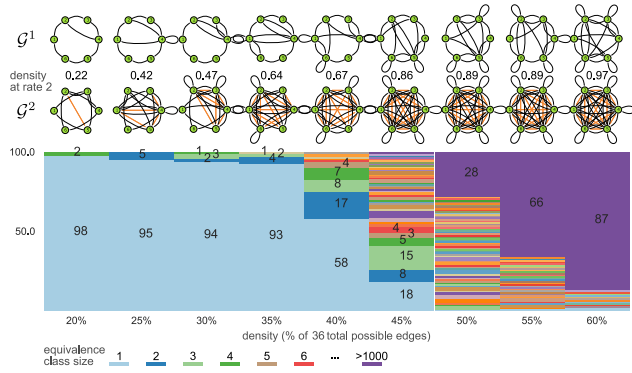


Figure 5: Equivalence class size distribution among 100 randomly generated 6-node SCCs at a given density and examples of 6-node SCCs for each.

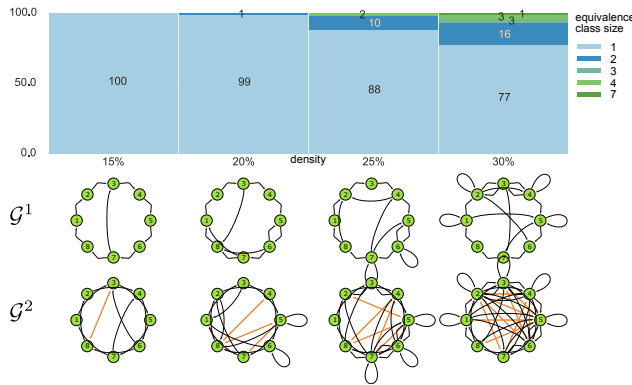


Figure 6: Equivalence class size distribution for 100 randomly generated 8-node SCCs at a given density and examples of 8-node SCCs for each of the densities.

100 random 6-node SCCs (using the above procedure) for each density from 20% to 60% in 5% increments, then analytically computed  $\mathcal{G}^2$  and passed that to the optimized MSL algorithm.  $n = 6$  is sufficiently large that brute-force inference is infeasible, but the graphs are still tractable for the optimized MSL algorithm even at high densities. This is a particular worry since the complexity of the search grows exponentially with the number of edges in  $\mathcal{G}^2$ . Figure 5 shows the sizes of the equivalence classes for the graphs at different densities.

Some of the most notable findings from Figure 5 are *i*) for densities up to 35%, the overwhelming majority of the equivalence classes are singletons; *ii*) for densities above 50%, the equivalence classes often grow to quite large sizes; however, *iii*) those  $\mathcal{G}^2$  graphs are incredibly dense, and so unsurprisingly are difficult to analyze tractably. The MSL algorithm complexity depends exponentially on the number of edges in  $\mathcal{G}^2$ , and so increasing  $n$  for a fixed density rapidly leads to significant computational barriers. Figure 5 suggests that densities above 35% will frequently lead to very large equivalence classes, and so we focus on lower

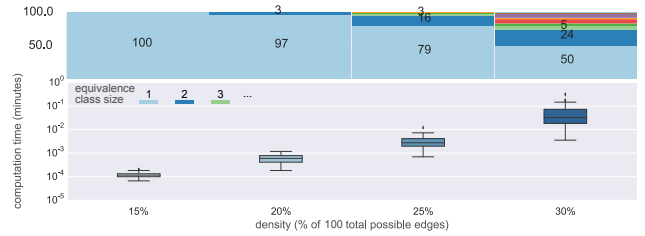


Figure 7: Distribution of run-time (wall clock) and sizes of equivalence classes across densities of 10-node graphs.

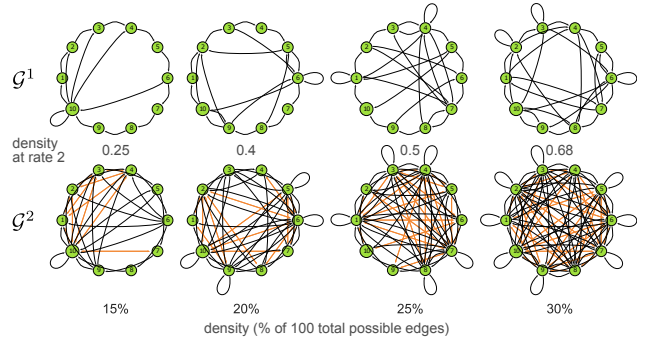


Figure 8: Example 10-node graphs at different densities and their corresponding  $\mathcal{G}^2$ s.

densities for larger  $n$ .

In particular, we performed the same analysis (including run-times) for 8-node (Figure 6) and 10-node graphs (Figure 7). The results for 6-node graphs largely generalize. MSL is a fast and practical algorithm for these graph sizes, as demonstrated by the wall-clock run-time measurements summarized in Figure 7. Note that these are quite challenging graphs, as shown in Figure 8. 10-node graphs with 30% density can have more than 65 edges in  $\mathcal{G}^2$ , and so the naive approach would have to consider  $10^{65}$  virtual node identifications.

The MSL algorithm is also computationally tractable for significantly larger  $n$ . For 15-node graphs, it can readily learn  $\mathcal{G}^1$  structures up to 25% density, though outlier cases can take multiple days to compute. Figure 9 shows results for 100s of random SCCs with density of 10% for node sizes from 15 to 35. This density actually corresponds to quite challenging learning tasks. For example, a 35-node graph with 10% density can have nearly 400 edges in  $\mathcal{G}^2$ . Despite these large numbers of edges, the MSL algorithm rarely takes longer than an hour, even for 35-node graphs.

## 4.2 GENERALITY OF THE MSL ALGORITHM

The MSL algorithm is actually an algorithm-schema that can be generalized to different known  $u$ , though its complexity rapidly increases. We performed a “proof-of-concept” of MSL for  $u = 3$  with 100 random 6-node graphs. In this variant, each  $\mathcal{G}^3$  edge has *two* virtual nodes

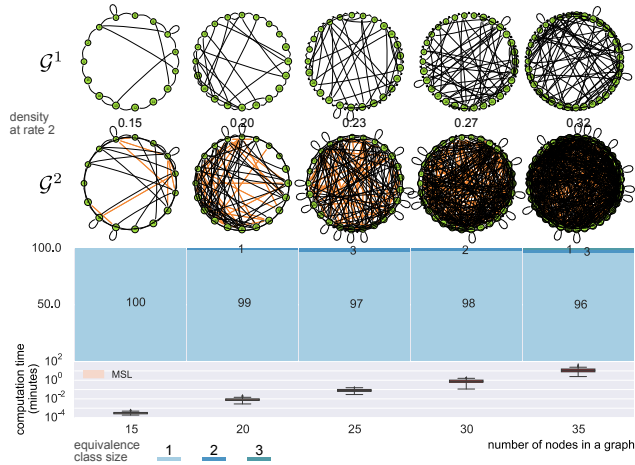


Figure 9: Graphs with 15, 20, 25, 30, and 35 nodes at the density of 10%, their corresponding  $\mathcal{G}^2$  and equivalence class size distribution as well as the running time summarizing the computation of 100 random SCCs per node size.

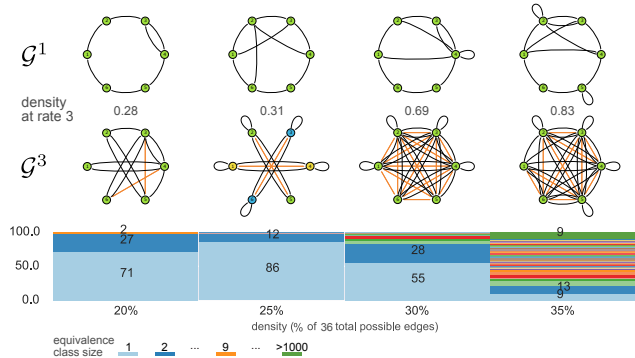


Figure 10: Applying the MSL algorithm to the  $\mathcal{G}^3 \rightarrow \mathcal{G}^1$  problem: results for 100 random 6 node graphs per density. Example  $\mathcal{G}^1$  and their corresponding  $\mathcal{G}^3$  are shown for reference.

that must be identified. Figure 10 shows the equivalence classes for  $u = 3$ . Interestingly, we obtain similar results with singleton equivalence classes dominating at low densities, but a rapid increase in the proportion of larger equivalence classes.

### 4.3 VIOLATIONS OF THE RATE ASSUMPTIONS

If we know that  $u = 2$ , then MSL can successfully recover all graphs in the equivalence class. That assumption could easily be violated, however, as one might (for example) believe  $u = 2$  when actually  $u = 3$ . Figure 11 shows results when MSL assuming  $u = 2$  was applied to the  $\mathcal{G}^3$  for 100 6-node graphs. Importantly, for  $\mathcal{G}^1$  densities up to 30%, the MSL (assuming  $u = 2$ ) algorithm fails to find a solution; that is, there is no  $\mathcal{G}^1$  whose  $\mathcal{G}^2$  matches the given graph (which is actually a  $\mathcal{G}^3$ ). One can thus infer that a key algorithmic assumption has likely been violated. Un-

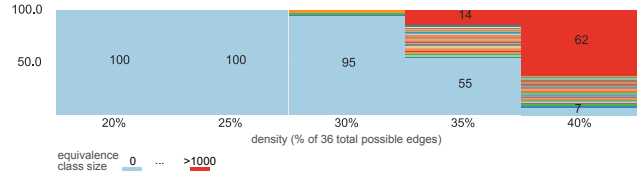


Figure 11: Equivalence class size distribution for 100 randomly generated 6-node SCCs at a given density after the  $\mathcal{G}^2 \rightarrow \mathcal{G}^1$  MSL search when the input graph to the algorithm was in fact  $\mathcal{G}^3$

fortunately, the testability of the  $u = 2$  assumption does not seem to extend to higher densities, which further suggests focusing our attention on  $\mathcal{G}^1$  with no more than 30% density.

### 4.4 FROM UNDERSAMPLED DATA TO $\mathcal{G}^1$

The above results show that the nonparametric  $\mathcal{G}^2 \rightarrow \mathcal{G}^1$  component of the MSL algorithm is correct and computationally efficient, enabling us to learn equivalence classes for SCCs with densities up to 30%. For finite sample data, we need to incorporate a stable, reliable algorithm to learn the  $\mathcal{H}^2$  structure. As noted earlier, there are multiple learning algorithms for this task (i.e., learning measurement-timescale structure), though they must allow for the possibility of bidirected edges to encode correlation between variables at the same time. We have used both a restricted version of the PC algorithm [11] and direct optimization of log likelihood in a structural vector autoregressive (SVAR) model [10]. In our experiments, we found that SVAR optimization provided more accurate and stable solutions, likely because of errors in conditional independence tests used in the modified PC algorithm. The best choice for learning  $\mathcal{H}^2$  structure is an open research question. In the following, we only show results for the SVAR procedure.

We generated 100 random 8-node SCCs for each density in  $\{15\%, 20\%, 25\%, 30\%\}$ . These graphs are complex and interesting, but also are computationally tractable for the full MSL algorithm. For each random graph, we generated a random transition matrix by sampling weights for the non-zero elements of the adjacency matrix, and controlling system stability (by keeping the maximal eigenvalue at or below 1). This transition matrix was then used for a vector auto-regressive (VAR) model [10] with noise (standard deviation of 1) to generate data. Every other data sample was removed (to undersample at rate 2), and the resulting data was provided to the SVAR optimization to yield a candidate  $\mathcal{H}^2$ . The MSL algorithm was then applied to this  $\mathcal{H}^2$  to obtain an equivalence class of  $\mathcal{G}^1$  that can be compared to ground truth in terms of two error-types: **omission error**: the number of omitted edges normalized to the total number of edges in the ground truth; **omission error**: number of edges not present in the ground truth nor-

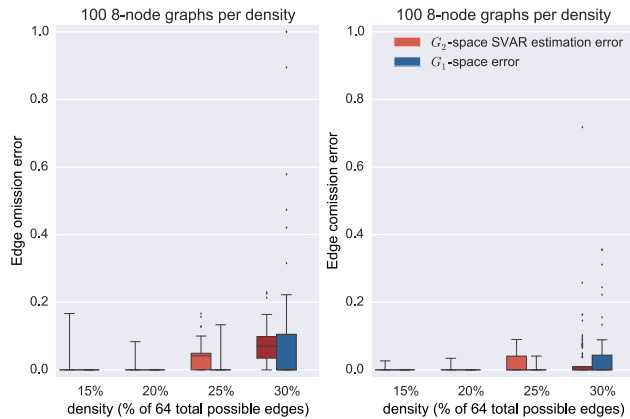


Figure 12: The MSL estimation and search errors on synthetic data undersampled at rate 2.

malized to the total possible edges minus the number of those present in the ground truth. Figure 12 shows the results of these simulations. We also plot the estimation errors of the SVAR (on the undersampled data) to understand the dependence of MSL estimation errors on the estimation errors for  $\mathcal{H}^2$ . Interestingly, applying the MSL algorithm does not significantly increase the error rates over those produced by the SVAR estimation.

In some cases, SVAR estimation errors result in an  $\mathcal{H}^2$  for which there are *no* possible  $\mathcal{G}^1$ .<sup>12</sup> For the simulations described in Figure 12, we deal with these cases by *i*) modifying MSL to accept (at the final step) those solutions that produce an undersampled graph that has the same directed edges as the estimated  $\mathcal{H}^2$ ; *ii*) restarting the simulation if a solution is not found. The former improves performance because bidirected edges often contain a weaker signal and are prone to mis-estimation, while the latter is to ensure comparability of results.

We have also modified the MSL algorithm so that, in these cases, it sequentially considers all neighbors of each  $\mathcal{H}^2$  in the Hamming cube constructed on the length  $n^2 + \binom{n}{2}$  binary string that represent directed and bidirected edges. If MSL finds a solution for one of these neighbors of  $\mathcal{H}^2$ , then we return it and compare to the ground truth as before.

We repeated our 8-node experiment successively checking neighborhoods from 1-5 steps away from the learned  $\mathcal{H}^2$ . In the worst case, this can require over  $5.2 \times 10^7$  additional MSL runs (for  $n = 8$ ), so there can be a significant increase in run-time. The results are summarized in Figure 13. The increased complexity did not allow us to proceed to the 30% density. This time, however, we did not have to restart a single computation at densities or 15% and 20% with only few rejected at 25%.

<sup>12</sup>Because the  $\mathcal{G}^1 \rightarrow \mathcal{G}^2$  map is many-to-one, there are multiple such  $\mathcal{H}^2$ . In fact, the set of “reachable”  $\mathcal{H}^2$  is at most  $1/2 \binom{n}{2}$  of the theoretically possible graphs.

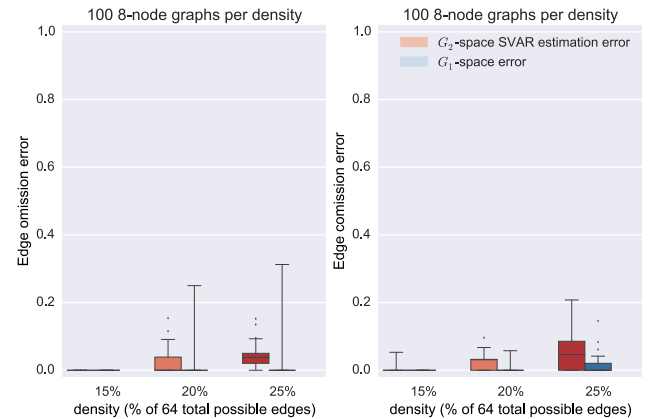


Figure 13: The estimation and search errors on synthetic data undersampled at rate 2 when the  $\mathcal{G}^2$  Hamming cube neighborhood search is used.

## 5 CONCLUSIONS

Many scientific contexts depend on learning the structure of a system at some timescale that is faster than the measurement timescale. Standard structure learning algorithms can extract the measurement-level structure, but that structure can be quite different from the structure of the underlying system. The apparent structure given undersampled data is not immediately informative about the actual structure at the causal or system timescale. We have presented the first computationally efficient algorithm for learning the equivalence class of system-timescale structures that could have produced the measurement-timescale data. The algorithm can, in theory, be applied for arbitrary known undersample rates  $u$ , though it is computationally intractable for  $u > 3$ . Nonetheless, we have shown that the MSL algorithm exhibits promising performance for  $u = 2$ , including reliably learning underlying structure over large node-sets. The MSL algorithm also provides a novel tool for investigating the sizes of those equivalence classes. We showed that small amounts of undersampling typically do not destroy much information, as the equivalence class for many  $\mathcal{G}^2$  was a singleton. Undersampling greatly increases the complexity of structure learning, but does not make it impossible or infeasible.

## ACKNOWLEDGEMENTS

Thanks to Kun Zhang for helpful conversations. SP & DD contributed equally. SP was supported by awards NIH R01EB005846 & NSF IIS-1318759. DD was supported by awards NSF IIS-1318815 & NIH U54HG008540 (from the National Human Genome Research Institute through funds provided by the trans-NIH Big Data to Knowledge (BD2K) initiative). The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.



## References

- [1] D. M. Chickering. Optimal structure identification with greedy search. *The Journal of Machine Learning Research*, 3:507–554, 2003.
- [2] D. Danks and S. Plis. Learning causal structure from undersampled time series. In *JMLR: Workshop and Conference Proceedings*, volume 1, pages 1–10, 2013.
- [3] D. Dash. Restructuring dynamic causal systems in equilibrium. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics (AISTats 2005)*, pages 81–88, 2005.
- [4] T. Dean and K. Kanazawa. A model for reasoning about persistence and causation. *Computational intelligence*, 5(2):142–150, 1989.
- [5] F. M Fisher. A correspondence principle for simultaneous equation models. *Econometrica: Journal of the Econometric Society*, pages 73–92, 1970.
- [6] N. Friedman, K. Murphy, and S. Russell. Learning the structure of dynamic probabilistic networks. In *15th Annual Conference on Uncertainty in Artificial Intelligence*, pages 139–147, San Francisco, 1999. Morgan Kaufmann.
- [7] C.W.J. Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica: Journal of the Econometric Society*, pages 424–438, 1969.
- [8] S. A. Huettel, A. W. Song, and G. McCarthy. *Functional magnetic resonance imaging*. Sinauer Associates, Publishers, Sunderland, MA, USA, 2004. ISBN 0-87893-288-7.
- [9] Y. Iwasaki and H. A Simon. Causality and model abstraction. *Artificial Intelligence*, 67(1):143–194, 1994.
- [10] H. Lütkepohl. *New introduction to multiple time series analysis*. Springer Science & Business Media, 2007.
- [11] A. Moneta, N. Chlaß, D. Entner, and P. Hoyer. Causal search in structural vector autoregressive models. In *Journal of Machine Learning Research: Workshop and Conference Proceedings, Causality in Time Series (Proc. NIPS2009 Mini-Symposium on Causality in Time Series)*, volume 12, pages 95–114, 2011.
- [12] K. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, UC Berkeley, 2002.
- [13] P. Spirtes, C. Glymour, and R. Scheines. *Causation, prediction, and search*, volume 81. MIT press, 2001.
- [14] B. Thiesson, D. Chickering, D. Heckerman, and C. Meek. ARMA time-series modeling with graphical models. In *Proceedings of the Twentieth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-04)*, pages 552–560, Arlington, Virginia, 2004. AUAI Press.
- [15] M. Voortman, D. Dash, and M. Druzdzel. Learning why things change: The difference-based causality learner. In *Proceedings of the Twenty-Sixth Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 641–650, Corvallis, Oregon, 2010. AUAI Press.

---

# Budgeted Online Collective Inference

---

**Jay Pujara**

University of Maryland  
jay@cs.umd.edu

**Ben London**

University of Maryland  
blondon@cs.umd.edu

**Lise Getoor**

University of California, Santa Cruz  
getoor@soe.ucsc.edu

## Abstract

Updating inference in response to new evidence is a fundamental challenge in artificial intelligence. Many real problems require large probabilistic graphical models, containing millions of interdependent variables. For such large models, jointly updating the most likely (i.e., MAP) configuration of the variables each time new evidence is encountered can be infeasible, even if inference is tractable. In this paper, we introduce *budgeted online collective inference*, in which the MAP configuration of a graphical model is updated efficiently by revising the assignments to a subset of the variables while holding others fixed. The goal is to selectively update certain variables without sacrificing quality with respect to full inference. To formalize the consequences of partially updating inference, we introduce the concept of *inference regret*. We derive inference regret bounds for a class of graphical models with strongly-convex free energies. These theoretical insights, combined with a thorough analysis of the optimization solver, motivate new approximate methods for efficiently updating the variable assignments under a budget constraint. In experiments, we demonstrate that our algorithms can reduce inference time by 65% with accuracy comparable to full inference.

## 1 INTRODUCTION

A key challenge of many artificial intelligence problems is that the evidence grows and changes over time, requiring updates to inferences. Every time a user rates a new movie on Netflix, posts a status update on Twitter, or adds a connection on LinkedIn, inferences about preferences, events, or relationships must be updated. When constructing a knowledge base, each newly acquired document prompts the system to update inferences over related facts and re-

solve mentions to their canonical entities. Problems such as these benefit from collective (i.e., joint) reasoning, but incorporating new evidence into a collective model is particularly challenging. New evidence can affect multiple predictions, so updating inference typically involves recomputing all predictions in an expensive global optimization. Even when a full inference update is tractable—which, using the best known methods, can be linear in the number of factors—it may still be impractical. For example, updating a knowledge graph with millions of facts can take hours (Pujara *et al.*, 2013), thereby requiring some compromise, either in the form of a deferment strategy or approximate update. In this work, we consider the task of efficiently updating the *maximum-a-posteriori* (MAP) state of a probabilistic graphical model, conditioned on evolving evidence. We refer to this problem as *online collective inference*.

In online collective inference, a single graphical model, describing the conditional distribution of a set of random variables with fixed dependency structure, is given. Over a series of epochs, the true assignments (i.e., labels) of certain variables are revealed, introducing new evidence with which we can update the assignments to the remaining unknowns. We constrain the problem by adding a budget, such that only a fixed percentage of variables can be updated in each epoch, necessitating some approximation to full inference. This constraint distinguishes our work from the vast body of literature on belief revision (e.g., Gärdenfors, 1992), Bayesian network updates (e.g., Buntine, 1991; Friedman and Goldszmidt, 1997; Li *et al.*, 2006), models for dynamic (Murphy, 2002) or sequential (Fine *et al.*, 1998) data, and adaptive inference (e.g., Acar *et al.*, 2008), which deal with exact updates to inference. We analyze budgeted online collective inference from both the theoretical and algorithmic perspectives, addressing two fundamental questions: How do we choose which variables to update? How “close” is the approximate inference update to the full inference update?

To formalize the latter question, we introduce the concept of *inference regret*. Informally, inference regret measures the amount of change induced by fixing (i.e., condi-

tioning on) certain variables in the inference optimization. We specifically analyze the inference regret of continuous graphical models whose inference objective is strongly convex. One instantiation of this class of models is hinge-loss Markov random fields (Bach *et al.*, 2013), which have been broadly applied and demonstrate state-of-the-art performance in many applications. Using the duality between strong convexity and stability, we upper-bound the inference regret. Our bound is proportional to the distance from the fixed variables to the optimal values of the full inference problem, scaled by a function of several model-specific properties. We use our inference regret bound to quantify the effect of approximate inference updates in response to new evidence (in this case, revealed labels). The bound highlights two terms affecting the regret: the accuracy of the original predictions and the amount that the original predictions change. This latter insight informs our approximate update methods with a simple intuition: fix the predictions that are unlikely to change in a full inference update.

To efficiently determine which variables are least likely to change, we turn to the optimization algorithm used during inference. The alternating direction method of multipliers (ADMM) (Boyd *et al.*, 2011) is a popular convex optimization technique that offers convergence guarantees while remaining highly scalable. We analyze the optimization process and catalog the features that allow us to determine which variables will change the most. Using these features to generate a score for each variable, we establish a ranking capturing the priority of including the variables in subsequent inference. Since the variable scores are produced using the state of the optimization algorithm, our method does not incur computational overhead. By ranking variables, we approximate full inference with an arbitrary budget and support an anytime online inference algorithm.

We evaluate the theoretical guarantees and approximation quality in experiments on a synthetic collective classification task and a real-world collaborative filtering task. These experiments validate our theoretical bounds by measuring the stability and quality of the MAP state as new evidence is revealed. To connect theoretical guarantees with empirical performance, we compare approximate inference to computing the full MAP state at each epoch of graph evolution. We find that our approach to online inference allows a substantial decrease in computation and running time while maintaining the quality of the inferred values. In our experiments, our methods consistently reduce running time by 65% to 70%, show diminishing inference regret, and, in some cases, have lower test error than full inference.

## 1.1 RELATED WORK

Updating inference is a longstanding problem in artificial intelligence. The classic problem of belief revision (Gardenfors, 1992) considers revising and updating a set of

propositional beliefs using a set of axiomatic guarantees to consistency. Diverse research has considered updating the parameters or structure of Bayesian networks in response to evolving evidence (Buntine, 1991; Friedman and Goldszmidt, 1997; Li *et al.*, 2006, e.g.). Finally, many models address dynamic or sequential data, such as Dynamic Bayesian Networks (Murphy, 2002) and hierarchical hidden Markov models (Fine *et al.*, 1998). Our work addresses the specific problem of approximating full MAP inference in the online setting when a model is given and provides formal guarantees for the approximation quality.

Making efficient updates to the full inference result is the goal of a related area of research, *adaptive inference*. Adaptive marginal inference (Acar *et al.*, 2008; Sümer *et al.*, 2011) can update the marginal probability of a query in  $O(2^{\text{tw}(G)} \log n)$ -time, where  $\text{tw}(G)$  is the tree-width of the graph and  $n$  is the number of variables. Adaptive MAP inference (Acar *et al.*, 2009) can update the MAP state in  $O(m + m \log(n/m))$ -time, where  $m$  is the number of variables that change their state. Though the algorithm does not need to know  $m$  beforehand, a model change could result in changes to all  $n$  variables’ states, with cost equivalent to exact inference. These adaptive inference techniques do not currently support partial updates to the MAP state or accommodate budgeted updates.

Approximate adaptive inference was considered by Nath and Domingos (2010), who proposed *expanding frontier belief propagation* (EFBP), a belief propagation algorithm that only updates messages in the vicinity of the updated potentials. They showed that the beliefs generated by EFBP lower- and upper-bound the beliefs of full BP, thereby providing guarantees on the quality of the approximation. This result differs from ours in that it bounds the individual marginal probabilities, whereas we bound the  $L^1$  distance between MAP states. Unlike our approximation algorithm, EFBP does not explicitly limit computation and, in the worst case, may need to update all variables to achieve convergence conditions.

The quantity we call inference regret is conceptually similar to *collective stability* (London *et al.*, 2013a). Collective stability measures the amount of change in the output of a structured predictor induced by local perturbations of the evidence. London *et al.* (2013a, 2014) analyzed the collective stability of marginal inference in discrete graphical models, concluding that (approximate) inference with a strongly convex entropy function enhances stability. Our technical approach is similar, in that we also leverage strong convexity. However, the types of perturbations we consider—fixing target variables—are not covered by their analysis. Stability analysis is closely related to *sensitivity analysis*. Since the terms are used interchangeably in the literature, we distinguish them as follows: sensitivity analysis examines *if* and *when* the solution changes; stability analysis examines *how much* it changes by. Laskey

analyzed the sensitivity of queries (which can be used for marginal inference) in Bayesian networks. Chan and Darwiche studied the sensitivity of queries (2005) and MAP inference (2006) in Markov networks. Their 2005 paper also analyzes the stability of queries.

## 2 PRELIMINARIES

The theory and methods introduced in this paper apply to any continuous-valued MRF with a strongly convex MAP inference objective function. One case of particular interest is a class of graphical models known as *hinge-loss Markov random fields* (HL-MRFs) (Bach et al., 2013). An HL-MRF is a continuous-valued Markov network in which the potentials are hinge functions of the variables. Our choice of HL-MRFs comes from technical considerations: we reason about the strength of convexity of the inference objective, and *maximum a posteriori* (MAP) inference in HL-MRFs can be strongly convex. However, from a practical standpoint, HL-MRFs have many benefits. MAP inference in HL-MRFs is provably and empirically efficient, in theory growing  $O(N^3)$  with the number of potentials,  $N$ , but in practice often converging in  $O(N)$  time. Models built using HL-MRFs achieve state-of-the-art performance for a variety of applications (Bach et al., 2013; Beltagy et al., 2014; Chen et al., 2014; Fakhraei et al., 2014; London et al., 2013b; Ramesh et al., 2014). Finally, HL-MRFs are easily specified through *probabilistic soft logic* (PSL) (Bach et al., 2015), a probabilistic programming language with a first-order logical syntax.

To better understand HL-MRFs and PSL, consider a model for collective classification of network data, in which the goal is to assign labels to nodes, conditioned on some local evidence and network structure. Let  $G \triangleq (\mathcal{V}, \mathcal{E})$  denote an undirected graph on  $n \triangleq |\mathcal{V}|$  nodes. Each node  $i \in \mathcal{V}$  is associated with a set of local observations,  $X_i$ , and an unknown label,  $L_i$ . (In some settings, a subset of the labels are revealed.) In general, the observations and labels can be real-valued; but for simplicity of exposition, let us assume that each observation is binary-valued, and each label is categorical. The following logical rules define a PSL program for a typical collective classification model:

$$w_{x,\ell} : \text{FEATURE}(N, x) \Rightarrow \text{LABEL}(N, \ell)$$

$$w_{e,\ell} : \text{EDGE}(N_1, N_2) \wedge \text{LABEL}(N_1, \ell) \Rightarrow \text{LABEL}(N_2, \ell)$$

Variables  $N$ ,  $N_1$  and  $N_2$  denote nodes;  $x$  indexes a local feature; and  $\ell$  denotes a label. The rules are weighted by  $w_{x,\ell}$  and  $w_{e,\ell}$  respectively. Given  $G$  and  $\mathbf{X} \triangleq (X_1, \dots, X_n)$  (and possibly some subset of the labels), the rules are *grounded out* for all possible instantiations of the predicates. The groundings involving unknown variables—in this case, groundings of the LABEL predicate—are represented by  $[0, 1]$ -valued variables,  $\mathbf{Y} \triangleq (Y_{i,\ell})_{i,\ell}$ . Using a relaxation of the MAX-SAT problem to continuous do-

main (Globerson and Jaakkola, 2007), each grounding is converted to a convex hinge function of the form

$$f(\mathbf{X}, \mathbf{Y}) = (\max\{0, \varphi(\mathbf{X}, \mathbf{Y})\})^q,$$

where  $\varphi$  is a linear function of  $(\mathbf{X}, \mathbf{Y})$ , and  $q \in \{1, 2\}$  is an exponent that is set *a priori* for the given rule. Each hinge function becomes a potential in an HL-MRF.

The resulting HL-MRF enables probabilistic inference over the set of PSL rules. Fix a set of  $r$  PSL rules, with corresponding weights  $\mathbf{w} \triangleq (w_1, \dots, w_r)$ . For the  $i^{\text{th}}$  rule, let  $\mathcal{G}(i)$  denote its set of groundings in  $G$ , and let  $f_j^i$  denote the  $j^{\text{th}}$  grounding of its associated hinge function. To compactly express the weighted sum of grounded rules, we let

$$\mathbf{f}(\mathbf{X}, \mathbf{Y}) \triangleq \left[ \sum_{j=1}^{|\mathcal{G}(1)|} f_j^1(\mathbf{X}, \mathbf{Y}), \dots, \sum_{j=1}^{|\mathcal{G}(r)|} f_j^r(\mathbf{X}, \mathbf{Y}) \right]^\top$$

denote the aggregate of the grounded hinge functions. We can thus write the weighted sum of groundings as  $\mathbf{w} \cdot \mathbf{f}(\mathbf{X}, \mathbf{Y})$ . This inner product defines a distribution over  $(\mathbf{Y} | \mathbf{X})$  with probability density function  $p(\mathbf{Y} = \mathbf{y} | \mathbf{X} = \mathbf{x}; \mathbf{w}) \propto \exp(-\mathbf{w} \cdot \mathbf{f}(\mathbf{X}, \mathbf{Y}))$ . The maximizer of the density function (alternately, the minimizer of  $-\mathbf{w} \cdot \mathbf{f}(\mathbf{X}, \mathbf{Y})$ ) is the MAP state. The values of  $\mathbf{Y}$  in the MAP state can be interpreted as confidences. Additionally, we can define a prior distribution over each  $\mathbf{Y}$ . In this case, we will use an  $L^2$ , or Gaussian, prior. This can be accomplished using the rule  $w_{p,\ell} : \neg \text{LABEL}(N, \ell)$ , with a squared hinge (i.e.,  $q = 2$ ). Let us assume, without loss of generality, that each prior rule has weight  $w_{p,\ell} = w_p/2$ , for some  $w_p > 0$ . Thus, the corresponding hinge function for grounding LABEL( $i, \ell$ ) is simply  $(Y_{i,\ell})^2$ ; the aggregate features for the prior are  $\|\mathbf{Y}\|_2^2$ . So as to simplify notation, let  $\dot{\mathbf{w}} \triangleq (\mathbf{w}, w_p)$  and define an *energy* function,

$$E(\mathbf{y} | \mathbf{x}; \dot{\mathbf{w}}) \triangleq \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, \mathbf{y}) + \frac{w_p}{2} \|\mathbf{y}\|_2^2. \quad (1)$$

The resulting probability density function is

$$p(\mathbf{Y} = \mathbf{y} | \mathbf{X} = \mathbf{x}; \dot{\mathbf{w}}) \propto \exp(-E(\mathbf{y} | \mathbf{x}; \dot{\mathbf{w}})).$$

MAP inference, henceforth denoted  $h(\mathbf{x}; \dot{\mathbf{w}})$ , is given by

$$h(\mathbf{x}; \dot{\mathbf{w}}) = \arg \min_{\mathbf{y}} E(\mathbf{y} | \mathbf{x}; \dot{\mathbf{w}}).$$

## 3 INFERENCE REGRET

The notion of *regret* has often been used to measure the loss incurred by an online learning algorithm relative to the optimal hypothesis. We extend this concept to online inference. Fix a model. Suppose we are given evidence,  $\mathbf{X} = \mathbf{x}$ , from which we make a prediction,  $\mathbf{Y} = \mathbf{y}$ , using MAP inference. Then, some subset of the unknowns are

revealed. Conditioning on the new evidence, we have two choices: we can recompute the MAP state of the remaining variables, using full inference; or, we can fix some of the previous predictions, and only update a certain subset of the variables. To understand the consequences of fixing our previous predictions we must answer a basic question: how much have the old predictions changed?

We formalize the above question in the following concept.

**Definition 1.** Fix a budget  $m \geq 1$ . For some subset  $\mathcal{S} \subseteq \{1, \dots, n\}$ , such that its complement  $\bar{\mathcal{S}} \triangleq \{1, \dots, n\} \setminus \mathcal{S}$ , has size  $|\bar{\mathcal{S}}| = m$ , let  $\mathbf{Y}_{\mathcal{S}}$  denote the corresponding subset of the variables, and let  $\mathbf{Y}_{\bar{\mathcal{S}}}$  denote its complement. Assume there is an operator  $\Gamma$  that concatenates  $\mathbf{Y}_{\mathcal{S}}$  and  $\mathbf{Y}_{\bar{\mathcal{S}}}$  in the correct order. Fix a model,  $\hat{\mathbf{w}}$ , and an observation,  $\mathbf{X} = \mathbf{x}$ . Further, fix an assignment,  $\mathbf{Y}_{\mathcal{S}} = \mathbf{y}_{\mathcal{S}}$ , and let

$$h(\mathbf{x}, \mathbf{y}_{\mathcal{S}}; \hat{\mathbf{w}}) \triangleq \Gamma \left( \mathbf{y}_{\mathcal{S}}, \arg \min_{\mathbf{y}_{\bar{\mathcal{S}}}} E(\Gamma(\mathbf{y}_{\mathcal{S}}, \mathbf{y}_{\bar{\mathcal{S}}}) | \mathbf{x}; \hat{\mathbf{w}}) \right)$$

denote the new MAP configuration for  $\mathbf{Y}_{\bar{\mathcal{S}}}$  after fixing  $\mathbf{Y}_{\mathcal{S}}$  to  $\mathbf{y}_{\mathcal{S}}$ . We define the *inference regret* for  $(\mathbf{x}, \mathbf{y}_{\mathcal{S}}; \hat{\mathbf{w}})$  as

$$\mathfrak{R}_n(\mathbf{x}, \mathbf{y}_{\mathcal{S}}; \hat{\mathbf{w}}) \triangleq \frac{1}{n} \|h(\mathbf{x}; \hat{\mathbf{w}}) - h(\mathbf{x}, \mathbf{y}_{\mathcal{S}}; \hat{\mathbf{w}})\|_1. \quad (2)$$

In general, the inference regret can be as high as 1 for variables in  $[0, 1]$ . For example, consider network classification model in which probability mass is only assigned to configurations where all nodes have the same label. Fixing a variable corresponding to a single node label in this setting is tantamount to fixing the label for all nodes. In the presence of strong evidence for a different label, incorrectly fixing a single variable results in incorrectly inferring all variables.

In online inference, regret can come from two sources. First, there is the regret of not updating the MAP state given new evidence (in this case, revealed labels). If this regret is low, it may not be worthwhile to update inference, which can be useful in situations where updating inference is expensive (such as updating predicted attributes for all users in a social network). The second type of regret is from using an approximate inference update in which only certain variables are updated, while the rest are kept fixed to their previous values. We describe several such approximations in Section 4. In practice, one may have both types of regret, caused by approximate updates in response to new evidence. Note that the inference regret obeys the triangle inequality, so one can upper-bound the compound regret of multiple updates using the regret of each update.

### 3.1 REGRET BOUNDS FOR STRONGLY CONVEX INFERENCE

A convenient property of the  $L^2$  prior is that it is *strongly convex*, by which we mean the following.

**Definition 2.** Let  $\Omega \subseteq \mathbb{R}^n$  denote a convex set. A differentiable function,  $f : \Omega \rightarrow \mathbb{R}$ , is  $\kappa$ -strongly convex (w.r.t. the 2-norm) if, for all  $\boldsymbol{\omega}, \boldsymbol{\omega}' \in \Omega$ ,

$$\frac{\kappa}{2} \|\boldsymbol{\omega} - \boldsymbol{\omega}'\|_2^2 + \langle \nabla f(\boldsymbol{\omega}), \boldsymbol{\omega}' - \boldsymbol{\omega} \rangle \leq f(\boldsymbol{\omega}') - f(\boldsymbol{\omega}). \quad (3)$$

Strong convexity has a well-known duality with stability, which we will use in our theoretical analysis.

The function  $f(\boldsymbol{\omega}) \triangleq \frac{1}{2} \|\boldsymbol{\omega}\|_2^2$  is 1-strongly convex. Therefore, the prior,  $\frac{w_p}{2} \|\mathbf{y}\|_2^2$ , is at least  $w_p$ -strongly convex. We also have that the aggregated hinge functions,  $\mathbf{f}(\mathbf{x}, \mathbf{y})$ , are convex functions of  $\mathbf{Y}$ . Thus, it is easily verified that the energy,  $E(\mathbf{y} | \mathbf{x}; \hat{\mathbf{w}})$ , is at least a  $w_p$ -strongly convex function of  $\mathbf{y}$ . This yields the following upper bound on the inference regret.

**Proposition 1.** Fix a model with weights  $\hat{\mathbf{w}}$ . Assume there exists a constant  $B \in [0, \infty)$  such that, for any  $\mathbf{x}$ , and any  $\mathbf{y}, \mathbf{y}'$  that differ at coordinate  $i$ ,

$$\|\mathbf{f}(\mathbf{x}, \mathbf{y}) - \mathbf{f}(\mathbf{x}, \mathbf{y}')\|_2 \leq B |y_i - y'_i|. \quad (4)$$

Then, for any observations  $\mathbf{x}$ , any budget  $m \geq 1$ , any subset  $\mathcal{S} \subseteq \{1, \dots, n\} : |\bar{\mathcal{S}}| = m$ , and any assignments  $\mathbf{y}_{\mathcal{S}}$ , with  $\hat{\mathbf{y}} \triangleq h(\mathbf{x}; \hat{\mathbf{w}})$ , we have that

$$\mathfrak{R}_n(\mathbf{x}, \mathbf{y}_{\mathcal{S}}; \hat{\mathbf{w}}) \leq \sqrt{\frac{1}{n} \left( \frac{3}{2} + \frac{B \|\mathbf{w}\|_2}{w_p} \right) \|\mathbf{y}_{\mathcal{S}} - \hat{\mathbf{y}}_{\mathcal{S}}\|_1}.$$

**Proof** Due to space restrictions, the proof is somewhat abbreviated. Let  $\hat{\mathbf{y}} \triangleq h(\mathbf{x}; \hat{\mathbf{w}})$  denote the original MAP configuration, i.e., the minimizer of  $E(\cdot | \mathbf{x}; \hat{\mathbf{w}})$ . Let  $\hat{\mathbf{y}}' \triangleq h(\mathbf{x}, \mathbf{y}_{\mathcal{S}}; \hat{\mathbf{w}})$  denote the updated MAP state after conditioning, and note that  $\hat{\mathbf{y}}'_{\bar{\mathcal{S}}}$  is the minimizer of  $E(\Gamma(\mathbf{y}_{\mathcal{S}}, \cdot) | \mathbf{x}; \hat{\mathbf{w}})$ . Since  $\hat{\mathbf{y}}_{\mathcal{S}}$  may be different from  $\mathbf{y}_{\mathcal{S}}$ , we have that  $\hat{\mathbf{y}}$  may not be in the domain of  $E(\Gamma(\mathbf{y}_{\mathcal{S}}, \cdot) | \mathbf{x}; \hat{\mathbf{w}})$ . We therefore define a vector  $\tilde{\mathbf{y}} \in [0, 1]^n$  that is in the domain, and has minimal Hamming distance to  $\hat{\mathbf{y}}$ . Let  $\tilde{y}_i \triangleq y_i$  for all  $i \in \mathcal{S}$ , and  $\tilde{y}_j \triangleq \hat{y}_j$  for all  $j \notin \mathcal{S}$ . It can be shown that

$$\|\hat{\mathbf{y}}' - \hat{\mathbf{y}}\|_2^2 = \|\hat{\mathbf{y}}' - \tilde{\mathbf{y}}\|_2^2 + \|\tilde{\mathbf{y}} - \hat{\mathbf{y}}\|_2^2. \quad (5)$$

Further, since the domain of each  $Y_i$  is  $[0, 1]$ ,

$$\|\tilde{\mathbf{y}} - \hat{\mathbf{y}}\|_2^2 = \|\mathbf{y}_{\mathcal{S}} - \hat{\mathbf{y}}_{\mathcal{S}}\|_2^2 \leq \|\mathbf{y}_{\mathcal{S}} - \hat{\mathbf{y}}_{\mathcal{S}}\|_1. \quad (6)$$

Therefore, combining Equations 5 and 6,

$$\begin{aligned} \|\hat{\mathbf{y}} - \hat{\mathbf{y}}'\|_2^2 &= \frac{1}{2} \left( \|\hat{\mathbf{y}} - \hat{\mathbf{y}}'\|_2^2 + \|\hat{\mathbf{y}}' - \hat{\mathbf{y}}\|_2^2 \right) \\ &\leq \frac{1}{2} \left( \|\hat{\mathbf{y}} - \hat{\mathbf{y}}'\|_2^2 + \|\hat{\mathbf{y}}' - \tilde{\mathbf{y}}\|_2^2 + \|\mathbf{y}_{\mathcal{S}} - \hat{\mathbf{y}}_{\mathcal{S}}\|_1 \right). \end{aligned} \quad (7)$$

For any  $\kappa$ -strongly convex function,  $\varphi : \Omega \rightarrow \mathbb{R}$ , where  $\hat{\boldsymbol{\omega}} = \arg \min_{\boldsymbol{\omega} \in \Omega} \varphi(\boldsymbol{\omega})$  is the minimizer, then  $\forall \boldsymbol{\omega}' \in \Omega$ ,

$$\frac{1}{2} \|\hat{\boldsymbol{\omega}} - \boldsymbol{\omega}'\|_2^2 \leq \frac{1}{\kappa} (\varphi(\boldsymbol{\omega}') - \varphi(\hat{\boldsymbol{\omega}})). \quad (8)$$

Applying this identity to the first two terms in Equation 7, since  $E(\cdot | \mathbf{x}; \hat{\mathbf{w}})$  is  $w_p$ -strongly convex, we have that

$$\begin{aligned} & \frac{1}{2} \|\hat{\mathbf{y}} - \hat{\mathbf{y}}'\|_2^2 + \frac{1}{2} \|\hat{\mathbf{y}}' - \tilde{\mathbf{y}}\|_2^2 \\ & \leq \frac{1}{w_p} (E(\tilde{\mathbf{y}} | \mathbf{x}; \hat{\mathbf{w}}) - E(\hat{\mathbf{y}} | \mathbf{x}; \hat{\mathbf{w}})). \end{aligned} \quad (9)$$

The  $E(\hat{\mathbf{y}}' | \mathbf{x}; \hat{\mathbf{w}})$  terms cancel out. Expanding  $E(\cdot | \mathbf{x}; \hat{\mathbf{w}})$ ,

$$\begin{aligned} & E(\tilde{\mathbf{y}} | \mathbf{x}; \hat{\mathbf{w}}) - E(\hat{\mathbf{y}} | \mathbf{x}; \hat{\mathbf{w}}) \\ & = \mathbf{w} \cdot (\mathbf{f}(\mathbf{x}, \tilde{\mathbf{y}}) - \mathbf{f}(\mathbf{x}, \hat{\mathbf{y}})) + \frac{w_p}{2} (\|\tilde{\mathbf{y}}\|_2^2 - \|\hat{\mathbf{y}}\|_2^2) \\ & \leq \|\mathbf{w}\|_2 \|\mathbf{f}(\mathbf{x}, \tilde{\mathbf{y}}) - \mathbf{f}(\mathbf{x}, \hat{\mathbf{y}})\|_2 + w_p \|\mathbf{y}_S - \hat{\mathbf{y}}_S\|_1. \end{aligned} \quad (10)$$

The last inequality uses Cauchy-Schwarz and

$$\|\tilde{\mathbf{y}}\|_2^2 - \|\hat{\mathbf{y}}\|_2^2 \leq 2 \|\mathbf{y}_S - \hat{\mathbf{y}}_S\|_1.$$

Finally, we construct a series of vectors, indexed by each  $i \in \mathcal{S}$ , that transform  $\hat{\mathbf{y}}$  into  $\tilde{\mathbf{y}}$ , one coordinate at a time. For the following, let  $\mathcal{S}(j)$  denote the  $j^{\text{th}}$  element in  $\mathcal{S}$ . First, let  $\tilde{\mathbf{y}}^{(0)} \triangleq \hat{\mathbf{y}}$ ; then, for  $j = 1, \dots, m$ , let  $\tilde{\mathbf{y}}^{(j)}$  be equal to  $\tilde{\mathbf{y}}^{(j-1)}$  with index  $\mathcal{S}(j)$  replaced with value  $\tilde{y}_{\mathcal{S}(j)}$ . Note that  $\tilde{\mathbf{y}}^{(m)} = \tilde{\mathbf{y}}$ . Using the triangle inequality, one can show that

$$\begin{aligned} \|\mathbf{f}(\mathbf{x}, \tilde{\mathbf{y}}) - \mathbf{f}(\mathbf{x}, \hat{\mathbf{y}})\|_2 & \leq \sum_{j=1}^m \left\| \mathbf{f}(\mathbf{x}, \tilde{\mathbf{y}}^{(j)}) - \mathbf{f}(\mathbf{x}, \tilde{\mathbf{y}}^{(j-1)}) \right\|_2 \\ & \leq B \|\mathbf{y}_S - \hat{\mathbf{y}}_S\|_1. \end{aligned} \quad (11)$$

The last inequality uses Equation 4, since  $\tilde{\mathbf{y}}^{(j)}$  and  $\tilde{\mathbf{y}}^{(j-1)}$  differ at a single coordinate,  $\mathcal{S}(j)$ . Combining Equations 7 and 9 to 11, we have that

$$\|\hat{\mathbf{y}} - \hat{\mathbf{y}}'\|_2^2 \leq \left( \frac{3}{2} + \frac{B \|\mathbf{w}\|_2}{w_p} \right) \|\mathbf{y}_S - \hat{\mathbf{y}}_S\|_1.$$

We then multiply both sides of the inequality by  $1/n$  and take the square root. Using  $\frac{1}{n} \|\hat{\mathbf{y}} - \hat{\mathbf{y}}'\|_1 \leq \frac{1}{\sqrt{n}} \|\hat{\mathbf{y}} - \hat{\mathbf{y}}'\|_2$  finishes the proof. ■

Proposition 1 states that the inference regret is proportional to the  $L^1$  distance from  $\mathbf{y}_S$  to  $\hat{\mathbf{y}}_S$ , multiplied by a model-dependent quantity,  $O\left(\frac{B \|\mathbf{w}\|_2}{n w_p}\right)$ . Later in this section, we discuss how to bound the features' Lipschitz constant,  $B$ , demonstrating that it is typically a small constant (e.g., 1). Thus, assuming  $\|\mathbf{w}\|_2$  is bounded from above, and the weight on the prior,  $w_p$ , is bounded from below, the model-dependent term should decrease with the number of variables,  $n$ . For variables bounded in  $[0, 1]$ , the Hamming distance upper-bounds the  $L^1$  distance. Using this identity, a pessimistic upper bound for the distance term is  $\|\mathbf{y}_S - \hat{\mathbf{y}}_S\|_1 \leq |\mathcal{S}|$ . In this case, the regret is proportional to  $O(\sqrt{|\mathcal{S}|/n})$ ; i.e., the square root of the fraction of the

variables that are fixed. While this yields a uniform, analytic upper bound, we gain more insight by considering the specific contexts.

For instance, suppose  $\mathbf{y}_S$  is a set of labels that has been revealed. Then  $\mathfrak{R}_n(\mathbf{x}, \mathbf{y}_S; \hat{\mathbf{w}})$  is the regret of not updating inference conditioned on new evidence, and  $\|\mathbf{y}_S - \hat{\mathbf{y}}_S\|_1$  is the  $L^1$  error of the original predictions w.r.t. the true labels. Now, suppose  $\mathbf{y}_S$  is a set of labels that are fixed from a previous round of inference. Then  $\mathfrak{R}_n(\mathbf{x}, \mathbf{y}_S; \hat{\mathbf{w}})$  is the regret of an approximate inference update, and  $\|\mathbf{y}_S - \hat{\mathbf{y}}_S\|_1$  is the  $L^1$  distance between the old predictions and the new predictions in the full inference update. Thus, to minimize this regret, we must fix values that are already close to what we think they will be in the updated MAP state. This criteria motivates our approximate update methods in Section 4.

### 3.1.1 The Lipschitz Constant of the Features

In this section, we give some intuition on how to bound the Lipschitz constant of the features,  $B$ , by considering a specific example. Suppose the model has a single rule:  $X \Rightarrow Y$ . The corresponding hinge is  $f(X, Y) \triangleq \max\{0, X - Y\}$ . Using the fact that  $|\max\{0, a\} - \max\{0, b\}| \leq |a - b|$ , one can show that  $\|\mathbf{f}(\mathbf{x}, \mathbf{y}) - \mathbf{f}(\mathbf{x}, \mathbf{y}')\|_2 \leq |y_i - y'_i| \leq 1$ , so  $B = 1$ .

PSL models typically use rules of this nature, with varying arity (i.e., diadic, triadic, etc.). In general,  $B$  should grow linearly with the number of groundings involving any single variable (i.e., the maximum degree of the factor graph). The number of groundings generated by each rule depends on its arity and the data. For instance, the relational rule in Section 2 will ground out once for each edge and each label; if there are 2 labels, and the maximum degree is bounded by a constant,  $\Delta$ , then the number of groundings generated by this rule for any single variable is at most  $2\Delta$ . Thus, in many practical models,  $B$  will be a small constant.

## 4 BUDGETED ONLINE INFERENCE

The bounds presented in Section 3.1 suggest that online collective inference under budget constraints is close to the full inference update when one is able to successfully choose and fix variables whose inferred values will have little or no change. We refer to the complementary process of selecting which variables to infer as *activation*. In practice, designing an activation algorithm is difficult. The optimization problem required to choose a set of variables, each with heterogeneous regret and optimization cost, that do not exceed an optimization budget is an instance of the NP-hard knapsack problem. Given the intrinsic intractability of selecting an optimal set of variables, we present two algorithms that employ theoretical insights from the previous section and show promise in empirical experiments.

## 4.1 BACKGROUND: ADMM OPTIMIZATION

To develop activation algorithms, we turn to the optimization technique used to determine the MAP state in HL-MRFs. [Bach et al. \(2012\)](#) have shown that applying consensus optimization using the Alternating Direction Method of Multipliers (ADMM) ([Boyd et al., 2011](#)) provides scalable inference for HL-MRFs. For clearer exposition, we express the inference in terms of the set of ground rules,  $\mathcal{G}$  and rewrite the energy function in Section 2 as:

$$E(\mathbf{y} | \mathbf{x}; \dot{\mathbf{w}}) \triangleq \sum_{g \in \mathcal{G}} w_g f_g(\mathbf{x}, \mathbf{y}) + \frac{w_p}{2} \|\mathbf{y}\|_2^2$$

Here,  $w_g f_g(\mathbf{x}, \mathbf{y})$  is a weighted potential corresponding to a single ground rule. ADMM substitutes the global optimization problem with local optimizations for each potential using independent copies of the variables. For each grounding  $g \in \mathcal{G}$ , let  $\mathbf{y}_g$  denote the variables involved in  $g$  and  $\tilde{\mathbf{y}}_g$  indicate the local copy of those variables. To reconcile the local optimizations, ADMM introduces a constraint that local variable copies agree with the global “consensus” for each variable  $i$  involved in the grounding; that is,  $\mathbf{y}_g[i] = \tilde{\mathbf{y}}_g[i]$ . This constraint is transformed into an augmented Lagrangian with penalty parameter  $\rho > 0$  and Lagrange multipliers  $\alpha_g$ :

$$\min_{\tilde{\mathbf{y}}_g} w_g f_g(\mathbf{x}, \tilde{\mathbf{y}}_g) + \frac{\rho}{2} \left\| \tilde{\mathbf{y}}_g - \mathbf{y}_g + \frac{1}{\rho} \alpha_g \right\|^2 \quad (12)$$

ADMM iteratively alternates optimizing the local potentials, then updating the consensus estimates and associated Lagrange multipliers for each variable, as such:

$$\begin{aligned} \tilde{\mathbf{y}}_g &\leftarrow \operatorname{argmin}_{\tilde{\mathbf{y}}_g} w_g f_g(\mathbf{x}, \tilde{\mathbf{y}}_g) + \frac{\rho}{2} \left\| \tilde{\mathbf{y}}_g - \mathbf{y}_g + \frac{1}{\rho} \alpha_g \right\|^2; \\ \mathbf{y}[i] &\leftarrow \operatorname{mean}_g(\tilde{\mathbf{y}}_g[i]); \quad \alpha_g[i] \leftarrow \alpha_g[i] + \rho(\tilde{\mathbf{y}}_g[i] - \mathbf{y}_g[i]). \end{aligned}$$

A key element of this optimization is the interplay of two components: the weighted potential corresponding to a grounding and the Lagrangian penalty for deviating from the consensus estimate. As optimization proceeds, the Lagrange multipliers are updated to increase the penalty for deviating from the global consensus. At convergence, a balance exists between the two components, reconciling the local minimizer and the aggregate of global potentials.

## 4.2 ADMM FEATURES

The goal of activation is to determine which variables are most likely to change in a future inference. From the analysis in the previous section, we can identify several basic elements for each variable in the model that serve as features for an activation algorithm. For each variable, we have its value at convergence ( $\mathbf{y}[i]$ ), and for each grounding  $g$ , the weight ( $w_g$ ), the value of the potential ( $f_g(\mathbf{x}, \tilde{\mathbf{y}}_g)$ ), and the

Lagrange multipliers ( $\alpha_g[i]$ ) measuring the aggregate deviation from consensus. We discuss each of these features to motivate their importance in an activation algorithm.

The value of a variable at convergence can provide a useful signal in certain situations, where a model has clear semantics. For example, the formulation of HL-MRFs often lends itself to a logical interpretation with binary outcomes, as in the cases of collective classification of attributes that are either present or absent. In this setting, assignments in the vicinity of 0.5 represent uncertainty, and therefore provide good candidates for activation. Unfortunately, this feature is not universal. Many successful HL-MRF models adopt semantics that use continuous values to model continuous variables, such as pixel intensity in image completion tasks or Likert-scale ratings in recommender systems. In this case, the semantics of the variable’s consensus value may provide an ambiguous signal for activation.

The weighted potentials of each variable contribute directly to the probability of the MAP configuration. Since the log-probability is proportional to the *negated* energy,  $-E$ , high weights and high potential values decrease the probability of the assignment. Intuitively, activating those variables that contribute high weighted potentials provides the best mechanism for approaching the full inference MAP state. A complication to this approach is that each weighted potential can depend on many variables. However, the potential value is a scalar quantity and there is no general mechanism to apportion the loss to the contributing variables.

In contrast, the Lagrange multipliers provide a granular perspective on each variable’s effect on Equation 12. For each variable copy ( $\tilde{\mathbf{y}}_g$ ), the Lagrange multiplier aggregates the difference between the copy and the global consensus across iterations. High Lagrange multipliers signal discord between the local minimizer and the global minimizer, indicating volatility. Activating variables with high Lagrange multipliers can resolve this discord in future inference using updated evidence. However, updated evidence may also resolve the disagreement between the local and global minimum, obviating an update to the variable.

## 4.3 ACTIVATION ALGORITHMS

Building on our analysis of ADMM optimization, we introduce two activation algorithms for online collective inference, “agnostic activation” and “relational activation”. Both algorithms produce a ranking that prioritizes each variable for inclusion in inference. The key difference between these algorithms is whether new or updated evidence is an input to the algorithm. Agnostic activation scores variables concurrently with inference, based on their susceptibility to change in future inferences. In contrast, relational activation runs prior to inference, with scores based primarily on relationships between variables and updated evidence in the factor graph.

Each approach has different advantages. Agnostic activation scores variables during inference, providing a performance advantage since the scoring algorithm does not delay a future run of inference. However, this technique has a slower response to new evidence since scoring occurs before such evidence is available. Relational activation can respond to newly-arrived evidence and choose variables related to new evidence, but this requires delaying scoring which can add a computational overhead to inference.

Both activation algorithms output a ranking of the variables, which requires a scoring function. We introduce two scoring functions that use the ADMM features described Section 4.2. Our first scoring function, VALUE, captures the intuition that uncertain variables are valuable activation candidates using the function  $1 - |0.5 - \mathbf{y}[i]|$ , where  $\mathbf{y}[i]$  is the consensus value for variable  $i$ . The second scoring function, WLM, uses both the weight and Lagrange multipliers of each potential. For each variable, we define a set of weighted Lagrange multiplier magnitudes,  $\mathcal{A}_w[i] \triangleq \{|w_g \alpha_g[i]|\}$ . To obtain a single scalar score, we take the maximum value of  $\mathcal{A}_w[i]$ .

The agnostic activation algorithm simply ranks each variable by their score from a scoring function, irrespective of the new evidence. The RELATIONAL algorithm combines the score with information about the new evidence. Using the existing ground model, RELATIONAL first identifies all ground potentials dependent on the new evidence. Then, using these ground potentials as a seed set, the algorithm performs a breadth-first search of the factor graph adding the variables involved in each factor it encounters to the frontier. Traversing the factor graph can quickly identify many candidate variables, so we prioritize variables in the frontier by  $\frac{S}{2^d}$  where  $S$  is the score assigned by a scoring function and  $d$  is the minimum distance between the variable and an element of the seed set in the factor graph.

The ranking output by either agnostic or relational activation lets us prioritize which variables to activate. Given a budget for the number or percentage of variables to infer, we activate a corresponding number of variables from the ranking. The remaining variables are constrained to their previously inferred values. We selectively ground the model, including only those rules that involve an activated variable. Following inference on the ground model, we use the updated optimization state to produce new scores.

When an inactive variable is treated as a constant, it does not have any associated Lagrange multipliers, and lacks features for the WLM scoring function. Therefore, instead of treating fixed variables as constants, we introduce them as constrained variables in the optimization. This allows us to generate features by capturing the discrepancy between a variable’s constrained value and the value of its local copies in groundings involving activated variables.

Our implementation of the agnostic activation algorithm is

extremely efficient; all necessary features are byproducts of the inference optimization. Once scores are computed and the activated atoms are selected, the optimization state can be discarded to avoid additional resource commitments. In relational activation, scoring is similarly efficient, but there is an additional overhead of preserving the ground model to allow fast traversal of the factor graph. By selectively grounding the model, we replace queries that scan the entire database, potentially many times, with precise queries that exploit indices for faster performance. Finally, selectively activating atoms produces an optimization objective with fewer terms, allowing quicker optimization.

## 5 EVALUATION

To better understand the regret bounds and approximation algorithms for online inference, we perform an empirical evaluation on two online collective inference settings. The first setting is a synthetic online collective classification task where the data generator allows us to modulate the importance of collective dependencies and control the amount of noise. The second evaluation setting is a real-world collaborative filtering task, where user preferences are incrementally revealed and the outputs of a recommender system are correspondingly updated.

In both evaluation settings, we measure regret relative to full inference and inference error relative to ground truth. The results demonstrate that empirical regret follows the form of our regret bounds. We also evaluate the approximation algorithms presented in Section 4.3, to investigate whether features from the optimization algorithm can reliably determine which variables to activate. The results show that our approximation algorithms are able to reduce running time by upwards of 65%, with inference regret relative to full inference.

All experiments are implemented using the open-source PSL framework and our code is available on GitHub<sup>1</sup>.

### 5.1 ONLINE COLLECTIVE CLASSIFICATION

Our evaluation data simulates a collective classification problem of inferring labels for users in a social network as new evidence is incrementally revealed. Each user is assigned one of two mutually exclusive labels. Some portion of the users have observed labels, while the labels of the remaining users are inferred. At each epoch, the label of one more user is revealed, so the model must update the inferred labels for the remaining users with unknown labels.

For each user, we generate local and collective features correlated with the user’s label. Local features are generated for each user and label by drawing from a Gaussian distribution conditioned on the label, such that the mean is  $t$  for

<sup>1</sup><https://github.com/puuju/uai15-boci-code>



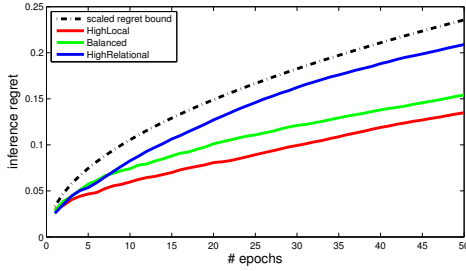


Figure 1: Inference regret, w.r.t. full inference, of fixing the original MAP state (i.e., no updates) in the HIGHLOCAL, HIGHCOLLECTIVE and BALANCED data models.

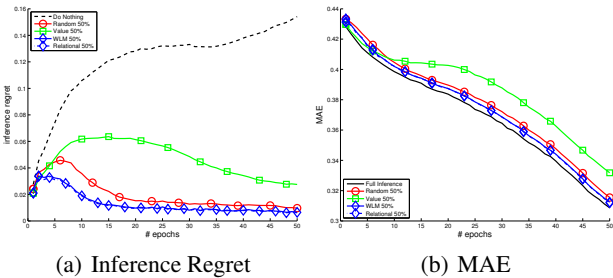


Figure 2: Inference regret (w.r.t. full inference) and MAE (w.r.t. ground truth) using various approximation algorithms, with 50% activation, in the COMPLEX data model.

the true label and  $1 - t$  for the incorrect label. The collective features are links between users, generated randomly using the following process: for each pair of users with the same label, a link is generated with probability  $p$ ; for each pair of users with different labels, a link is created with probability  $1 - p$ . We refer to  $p$  as the affinity of the network.

We model the data using the PSL rules described in Section 2 and learn weights for the model. Varying the parameters of the data generator impacts inference in the learned model, since the learned weights are proportional to the discriminative power of their associated rules. For example, varying the distance between the conditional means of the local features controls the importance of the local evidence rule: when the means are far apart, local evidence has high discriminative power; however, when the means are close, local evidence does not provide much signal.

We introduce three data models: HIGHLOCAL ( $t = .8, p = .75$ ), HIGHCOLLECTIVE ( $t = .55, p = .9$ ), and BALANCED ( $t = .7, p = .75$ ). We combine these three conditions in a fourth data model, COMPLEX, which samples uniformly from the three settings on a per-user basis resulting in heterogeneous evidence. For each condition, we generate 10 trials, each with a training social network used to learn the model parameters and a separate test social network to evaluate inference quality. Both the training and test graph have 100 users, with 60 observed user la-

els in the training graph and 10 observed user labels in the test graph. To infer user attributes, we use the simple collective classification model introduced in Section 2. We simulate the process of online inference by creating a sequence of observations consisting of 50 epochs. In each epoch, the true label of a previously unknown user is revealed, resulting in 60 observed user labels at the end of the sequence. For each trial, we generate 10 such sequences from a breadth-first traversal of the network from a randomly chosen user, resulting in a total of 5000 inferences.

In the first experiment, shown in Figure 1 we measure the inference regret of fixing variables to the initial MAP state (i.e., not updating inference) over 50 epochs, comparing the HIGHLOCAL, HIGHCOLLECTIVE and BALANCED conditions. Our theoretical analysis predicts that the worst-case regret grows at rate  $O(1/\sqrt{\text{epoch}})$ . The experimental results exhibit the same growth rate, which is very pronounced for the HIGHCOLLECTIVE data model, where variables are strongly interdependent, and less so for HIGHLOCAL, where variables are largely independent. The key insight is that the collective nature of the inference task determines the regret of online updates.

In the second experiment (Figure 2), we compare the approximate scoring algorithms with a budget of 50% of unknowns to running full inference on the COMPLEX network. We measure significance across 100 total sequences using a paired  $t$ -test with rejection threshold .05. For inference regret, we compare against the static algorithm, DONOTHING, which does not update the MAP state, and a random baseline, RANDOM, that fixes an arbitrary subset of 50% of the variables. We compare these to three approximation algorithms described in Section 4.1: VALUE, which uses the value assigned to the variable; WLM, which uses the maximum of the weighted Lagrange multipliers; and RELATIONAL, which uses WLM to prioritize exploration.

All methods exhibit low regret relative to full inference, contrasting the high regret of the static algorithm, although VALUE exhibits somewhat higher regret. The WLM and RELATIONAL methods have significantly lower regret relative to RANDOM, in 98% and 100% of epochs, respectively. We also compare the mean average error (MAE), with respect to ground truth, of using full inference vs. the approximations. This illustrates that the approximation algorithms remain competitive with full inference, although VALUE again lags in accuracy. Here, the WLM and RELATIONAL methods have significantly lower error than RANDOM in 80% and 100% of epochs, respectively. Comparing the running times highlights the computational benefit of using the approximation algorithms. The average running time for a single trial (which includes training and 10 random sequences of revealed variables) using full inference is 3076 seconds, while approximate inference requires only 955 seconds, a reduction of 69%, with inference time varying less than 3% across methods.

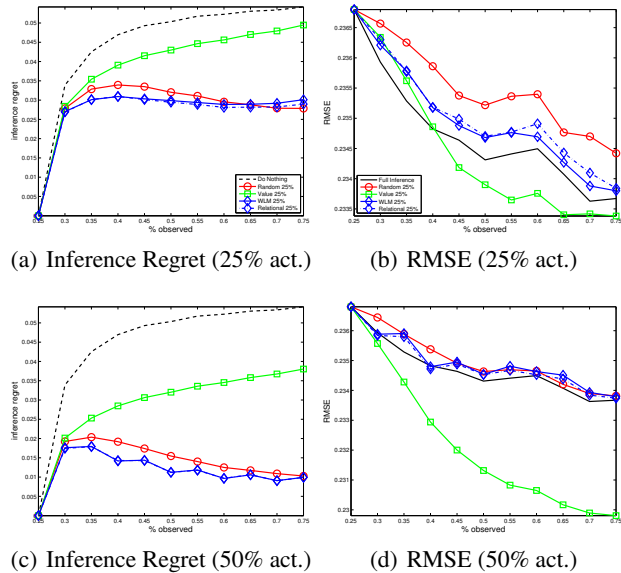


Figure 3: Inference regret (w.r.t. full inference) and RMSE (w.r.t. ground truth) for the Jester dataset.

## 5.2 COLLABORATIVE FILTERING

Our second evaluation task is a collaborative filtering task that employs a collective model to infer the preferences of users. We use the Jester dataset (Goldberg *et al.*, 2001) which includes ratings from 24,983 users on a set of 100 jokes. The task in this setting is to infer the user’s rating of each joke. We use the model from Bach *et al.* (2013) which assigns ratings to jokes based on the joke’s similarity to other jokes rated highly by the user. Joke similarity is measured using the mean-adjusted cosine similarity of the observed ratings of two jokes. (Refer to Bach *et al.* (2013) for further model details.) We sample 200 users who have rated all 100 jokes and split them into 100 training users and 100 testing users. We generate 10 sequences, each of which consists of a training and testing phase. Model weights are learned using 75% of the training users’ ratings observed. During testing, we incrementally reveal [25%, 30%, 40%, . . . , 75%] of the testing users’ ratings, performing online collective inference at each epoch.

We compare inference regret, relative to full inference, for the RANDOM, VALUE, WLM and RELATIONAL approximate methods. We also plot the RMSE, relative to ground truth, for full inference and all approximate methods. Figure 3a-b show results for 25% activation, and Figure 3c-d show 50% activation. Inference regret follows a similar pattern for both budgets, with VALUE showing increasing regret over epochs, and the remaining methods exhibiting level or diminishing regret after the first few epochs. The high regret for VALUE can be explained by considering the RMSE—VALUE actually *improves* the results of full inference, incurring high regret but low RMSE. Our intuition for

this improvement is that VALUE fixes polarized user ratings and allows these ratings to have greater influence on other unknown ratings, while full inference produces more moderate ratings for the entire set. The other approximation algorithms remain close to the full inference RMSE (at 50% activation) or perform slightly worse (at 25% activation). Comparing the running times, we find a similar improvement in speed. The average time for a sequence using full inference is 137 seconds, while the approximate methods require only 46 seconds, yielding a speedup of 66%. Approximation methods had consistent timing, varying less than 6%.

## 6 CONCLUSION

In this paper, we introduce a new problem, *budgeted online collective classification*, which addresses a common problem setting where online inference is necessary but full inference is infeasible, thereby requiring approximate inference updates. Our contributions are: (1) a formal analysis of online collective inference, introducing the concept of inference regret to measure the quality of the approximation; (2) analytic upper bounds on the inference regret incurred by strongly convex inference; and (3) several algorithms to address the practical problem of activation (i.e., choosing which variables to infer at each epoch), through a close analysis of the MAP inference optimization. Our empirical results demonstrate that our activation algorithms exhibit low inference regret and error that is competitive with full inference, while reducing the time required for inference by 65% or more.

This work inspires many exciting areas of future research. One open question is whether one can derive a tighter regret bound using the mechanics of the activation strategy, thus characterizing how performance degrades as a function of the budget. We are also interested in training an “optimal” activation policy that is trained using the variables whose values change the most during full inference. Finally, a crucial assumption in our analysis is that the model structure is fixed, but it is useful to consider the setting in which the set of variables change over time, allowing us to address situations such as new users joining a social network.

**Acknowledgments** This work was partially supported by National Science Foundation (NSF) grant IIS1218488 and by the Intelligence Advanced Research Projects Activity (IARPA) via DoI/NBC contract D12PC00337. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of NSF, IARPA, DoI/NBC, or the U.S. Government.

## References

- U. Acar, A. Ihler, R. Mettu, and Ö. Sümer. Adaptive inference on general graphical models. In *UAI*, 2008.
- U. Acar, A. Ihler, R. Mettu, and Ö. Sümer. Adaptive updates for MAP configurations with applications to bioinformatics. In *IEEE Statistical Signal Processing (SSP)*, pages 413–416. 2009.
- S. H. Bach, M. Broecheler, L. Getoor, and D. P. O’Leary. Scaling MPE inference for constrained continuous markov random fields with consensus optimization. In *NIPS*, 2012.
- S. H. Bach, B. Huang, B. London, and L. Getoor. Hinge-loss Markov random fields: Convex inference for structured prediction. In *UAI*, 2013.
- S. H. Bach, M. Broecheler, B. Huang, and L. Getoor. Hinge-loss Markov random fields and probabilistic soft logic. arXiv:1505.04406 [cs.LG], 2015.
- I. Beltagy, K. Erk, and R. J. Mooney. Probabilistic soft logic for semantic textual similarity. In *ACL*, 2014.
- S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends Machine Learning*, 3(1):1–122, 2011.
- W. Buntine. Theory refinement on bayesian networks. In *UAI*, 1991.
- H. Chan and A. Darwiche. Sensitivity analysis in Markov networks. In *IJCAI*, 2005.
- H. Chan and A. Darwiche. On the robustness of most probable explanations. In *UAI*, 2006.
- P.-T. Chen, F. Chen, and Z. Qian. Road traffic congestion monitoring in social media with hinge-loss Markov random fields. In *ICDM*, 2014.
- S. Fakhraei, B. Huang, L. Raschid, and L. Getoor. Network-based drug-target interaction prediction with probabilistic soft logic. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2014.
- S. Fine, Y. Singer, and N. Tishby. The hierarchical hidden Markov model: Analysis and applications. *Machine Learning*, 32(1):41–62, 1998.
- N. Friedman and M. Goldszmidt. Sequential update of Bayesian network structure. In *UAI*, 1997.
- Peter Gardenfors, editor. *Belief Revision*. Cambridge University Press, New York, NY, USA, 1992.
- A. Globerson and T. Jaakkola. Fixing max-product: convergent message passing algorithms for MAP LP-relaxations. In *NIPS*, 2007.
- K. Goldberg, T. Roeder, D. Gupta, and C. Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4(2):133–151, 2001.
- K. Laskey. Sensitivity analysis for probability assessments in Bayesian networks. In *UAI*, 1993.
- W. Li, P. van Beek, and P. Poupart. Performing incremental Bayesian inference by dynamic model counting. In *AAAI*, 2006.
- B. London, B. Huang, B. Taskar, and L. Getoor. Collective stability in structured prediction: Generalization from one example. In *ICML*, 2013.
- B. London, S. Khamis, S. H. Bach, B. Huang, L. Getoor, and L. Davis. Collective activity detection using hinge-loss markov random fields. In *CVPR Workshop on Structured Prediction: Tractability, Learning and Inference*, 2013.
- B. London, B. Huang, B. Taskar, and L. Getoor. PAC-Bayesian collective stability. In *AISTATS*, 2014.
- K. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, University of California, Berkeley, 2002.
- A. Nath and P. Domingos. Efficient belief propagation for utility maximization and repeated inference. In *AAAI*, 2010.
- J. Pujara, H. Miao, L. Getoor, and W. Cohen. Knowledge graph identification. In *ISWC*, 2013.
- A. Ramesh, D. Goldwasser, B. Huang, Hal Daumé III, and L. Getoor. Learning latent engagement patterns of students in online courses. In *AAAI*, 2014.
- Ö. Sümer, U. Acar, A. Ihler, and R. Mettu. Adaptive exact inference in graphical models. *JMLR*, 12:3147–3186, 2011.

---

# Auxiliary Gibbs Sampling for Inference in Piecewise-Constant Conditional Intensity Models

---

**Zhen Qin**

University of California, Riverside  
zqin001@cs.ucr.edu

**Christian R. Shelton**

University of California, Riverside  
cshelton@cs.ucr.edu

## Abstract

A piecewise-constant conditional intensity model (PCIM) is a non-Markovian model of temporal stochastic dependencies in continuous-time event streams. It allows efficient learning and forecasting given complete trajectories. However, no general inference algorithm has been developed for PCIMs. We propose an effective and efficient auxiliary Gibbs sampler for inference in PCIM, based on the idea of thinning for inhomogeneous Poisson processes. The sampler alternates between sampling a finite set of auxiliary virtual events with adaptive rates, and performing an efficient forward-backward pass at discrete times to generate samples. We show that our sampler can successfully perform inference tasks in both Markovian and non-Markovian models, and can be employed in Expectation-Maximization PCIM parameter estimation and structural learning with partially observed data.

## 1 INTRODUCTION

Modeling temporal dependencies in event streams has wide applications. For example, users' behaviors in online shopping and web searches, social network activities, and machines' responses in datacenter management can each be viewed as a stream of events over time. Models that can successfully learn the complex dependencies among events (both label and timing) allow targeted online advertising, automatic policy selection in datacenter management, user behavior modeling, or event prediction and dependency understanding in general.

[Gunawardana et al., 2011] proposed the piecewise-constant conditional intensity model (PCIM) which captures the dependencies among the types of events through a set of piecewise-constant conditional intensity

functions. A PCIM is represented as a set of decision trees, which allow for efficient model selection. Forecasting via forward sampling is also simple by iteratively sampling next events based on the current history.

However, currently model selection and forecasting for PCIMs is only effective given complete data. When there are missing data, an inference method is needed to answer general queries or be employed in expectation-maximization (EM) algorithms for model selection and parameter learning. Currently, no inference algorithm has been proposed for PCIM that can condition on general evidence.

In this work, we propose the first general inference algorithm for PCIMs, based on the idea of *thinning* for inhomogeneous Poisson process [Lewis and Shedler, 1979]. This results in an auxiliary Gibbs sampler that alternates between sampling a finite set of virtual event times given the current trajectory, and then sampling a new trajectory given the set of evidences and event times (virtual and actual). Our method is convergent, does not involve approximations like fixed time-discretization, and the samples generated can answer any type of query. We propose an efficient state-vector representation to maintain only necessary information for diverging trajectories, reducing the exponentially increasing sampling complexity to linear in most cases. We show empirically our inference algorithm converges to the true distribution, permits effective query answering, and aids model selection with incomplete data for PCIM models with both Markovian and complex non-Markovian dynamics. We also show the connection between PCIMs and continuous-time Bayesian networks (CTBNs), and compare our method with an existing method on such models.

## 2 PREVIOUS WORK

A dynamic Bayesian network (DBN) [Dean and Kanazawa, 1988] models temporal dependencies between variables in discrete time. For systems that evolve asynchronously without a global clock, it is

often not clear how timestamps should be discretized. Health records, computer server logs, and social networks are examples of asynchronous event data streams. For such systems, too slow a sampling rate would poorly represent the data, while too fast a sampling rate makes learning and inference more costly.

Continuous-time models have drawn attention recently in applications ranging from social networks [Du et al., 2013, Saito et al., 2009, Linderman and Adams, 2014] to genetics [Cohn et al., 2009] to biochemical networks [Golightly and Wilkinson, 2011]. Continuous Time Bayesian Networks (CTBN) [Nodelman et al., 2002] are homogeneous Markovian models of the joint trajectories of discrete finite variables, analogous to DBNs. Non-Markovian continuous models allow the rate of an event to be a function of the process’s history. Poisson Networks [Rajaram et al., 2005] constrain this function to depend only on the counts of the number of events during a finite time window. Poisson cascades [Simma and Jordan, 2010] define the rate function to be the sum of a kernel applied to each historic event, and requires the modeler to choose a parametric form for temporal dependencies.

A PCIM defines the intensity function as decision trees, with internal nodes’ tests mapping time and history to leaves. Each leaf is associated with a constant rate. A PCIM is able to model non-Markovian temporal dependencies, and is an order of magnitude faster to learn than Poisson networks. Applications include modeling super-computer event logs and forecasting future interests of web search users. While PCIMs have been extended in a number of ways [Parikh et al., 2012, Weiss and Page, 2013], there is no general inference algorithms.

Inference algorithms developed for continuous systems are mainly for Markovian models or specifically designed for a particular application. For CTBNs, there are variational approaches such as expectation propagation [El-Hay et al., 2010] and mean field [Cohn et al., 2009], which do not converge to the true value as computation time increases. Sampling based approaches include importance sampling [Fan et al., 2010] and Gibbs sampling [Rao and Teh, 2011, Rao and Teh, 2013] that converge to the true value. The latter is the current state-of-the-art method designed for general Markov Jump Processes (MJPs) and its extensions (including CTBNs). It uses the idea of uniformization [Grassmann, 1977] for Markov models, similar to thinning [Lewis and Shedler, 1979] for inhomogeneous Poisson processes. We note that our inference method generalizes theirs to non-Markovian models.

### 3 PCIM BACKGROUND

Assume events are drawn from a finite label set  $L$ . An event then can be represented by a time-stamp  $t$  and a label  $l$ . An event sequence  $x = \{(t_i, l_i)\}_{i=1}^n$ , where  $0 < t_1 <$

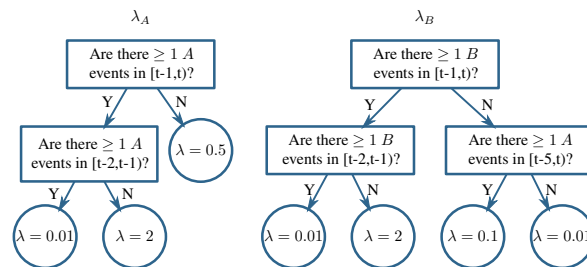


Figure 1: Decision tree representing  $S$  and  $\theta$  for events of labels  $A$  and  $B$ . Note the dependency among event labels (the rate of  $B$  depends on  $A$ ). [Gunawardana et al., 2011]

$\dots < t_n$ . We use  $h_i = \{(t_j, l_j) \mid (t_j, l_j) \in x, t_j < t_i\}$  for the history of event  $i$ , when it is clear from context which  $x$  is meant. We define the ending time  $t(y)$  of an event sequence  $y$  as the time of the last event in  $y$ , so that  $t(h_i) = t_{i-1}$ . A conditional intensity model (CIM) is a set of non-negative conditional intensity functions indexed by label  $\{\lambda_l(t|x; \theta)\}_{l=1}^{|L|}$ . The data likelihood is

$$p(x|\theta) = \prod_{l \in L} \prod_{i=1}^n \lambda_l(t_i|h_i; \theta)^{\mathbf{1}_{l(i)}} e^{-\Lambda_l(t_i|h_i; \theta)} \quad (1)$$

where  $\Lambda_l(t|h; \theta) = \int_{t(h)}^t \lambda_l(\tau|h; \theta) d\tau$ . The indicator function  $\mathbf{1}_{l'}$  is one if  $l' = l$  and zero otherwise.  $\lambda_l(t|h; \theta)$  is the expected rate of event  $l$  at time  $t$  given history  $h$  and model parameters  $\theta$ . Conditioning on the entire history causes the process to be non-Markovian. The modeling assumptions for a CIM are quite weak, as any distribution for  $x$  in which the timestamps are continuous random variables can be written in this form. Despite the weak assumptions, the per-label conditional factorization allows the modeling of label-specific dependence on past events.

A PCIM is a particular class of CIM that restricts  $\lambda(h)$  to be piecewise constant (as a function of time) for any history, so the integral for  $\Lambda$  breaks down into a finite number of components and forward sampling becomes feasible. A PCIM represents the conditional intensity functions with decision trees. Each internal node in a tree is a binary test of the history, and each leaf contains an intensity. If the tests are piecewise-constant functions of time for any event history, the resulting function  $\lambda(t|h)$  is piecewise-constant. Examples of admissible tests include

- Was the most recent event of label  $l$ ?
- Is the time of the day between 6am and 9am?
- Did an event with label  $l$  happen at least  $n$  times between 5 seconds ago and 2 seconds ago?
- Were the last two events of the same label?

Note some tests are non-Markovian in that they require knowledge of more than just which event was most recent. See Fig. 1 for an example of a PCIM model.

The decision tree for label  $l$  maps the time and history to a

leaf  $s \in \Sigma_l$ , where  $\Sigma_l$  is the set of leaves for  $l$ . The resulting data likelihood can be simplified:

$$p(x|S, \theta) = \prod_{l \in L} \prod_{s \in \Sigma_l} \lambda_{l_s}^{c_{l_s}(x)} e^{-\lambda_{l_s} d_{l_s}(x)}. \quad (2)$$

$S$  is the PCIM structure represented by the decision trees; the model parameters  $\theta$  are rates at the leaves.  $c_{l_s}(x)$  is the number of times label  $l$  occurs in  $x$  and is mapped to leaf  $s$ .  $d_{l_s}(x)$  is the total duration when the event trajectory for  $l$  is mapped to  $s$ .  $c$  and  $d$  are the sufficient statistics for calculating data likelihood.

[Gunawardana et al., 2011] showed that given the structure  $S$ , by using a product of Gamma distributions as a conjugate prior for  $\theta$ , the marginal likelihood of the data can be given in closed form, and thus parameter estimation can be done in closed form. Furthermore, imposing a structural prior allows a closed form Bayesian score to be used for greedy tree learning.

## 4 AUXILIARY GIBBS SAMPLING FOR PCIM

In this section we introduce our new inference algorithm for PCIM, called ThinnedGibbs, based on the idea of *thinning* for inhomogeneous Poisson processes. We handle incomplete data in which there are intervals of time during which events for particular label(s) are not observed.

### 4.1 Why Inference in PCIM is Difficult

Filling in partially observed trajectories for PCIM is hard due to the complex dependencies between unobserved events and both past and future events. See Fig. 2 for an example. While the history (the event at  $t$ ) says it is likely that there should be events in the unobserved area (with an expected rate of 2), future evidence (no events in  $R$ ) is contradictory: If there were indeed events in the unobserved area, those events should stimulate events happening in  $R$ .

Such a phenomenon might suggest existing algorithms such as the forward-filtering-backward-sampling (FFBS) algorithm for discrete-time Markov chains. However, there are two subtleties here: First, we are dealing with non-Markovian models. Second, we are dealing with continuous-time systems, so the number of time steps over which to propagate is infinite.

### 4.2 Thinning

Thinning [Lewis and Shedler, 1979] can be used to turn a continuous-time process into a discrete-time one, without using a fixed time-slice granularity. We select a rate  $\lambda^*$  greater than any in the inhomogeneous Poisson process and sample from a *homogeneous* process with this rate. To get a

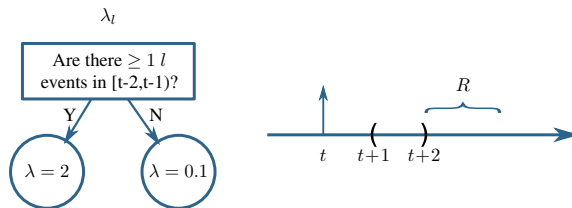


Figure 2: A simple PCIM with a partially observed trajectory. The vertical solid arrow indicates an evidence event. Areas between parentheses are unobserved. History alone indicates there should be events filled in, while the future (no events in  $R$ ) provides contradictory evidence.

sample from the original inhomogeneous process, an event at time  $t$  is thinned (dropped) with probability  $1 - \frac{\lambda(t)}{\lambda^*}$ .

This process can also be reversed. If given the set of thinned event times (samples from the inhomogeneous process), extra events can be added to a sample from the original constant-rate process by sampling from a Poisson process with rate  $\lambda^* - \lambda(t)$ . The cycle can then repeat by thinning the new total set of times (ignoring how they were generated). At each cycle, the times (after thinning) are drawn from the original inhomogeneous process. It is this type of cycle we will employ in our sampler.

The difficulty is a PCIM is not an inhomogeneous Poisson process. Its intensity depends on the entire history of events, not just the current time. For thinning, this means that we cannot independently sample whether each event is to be thinned. Furthermore, we wish to sample from the posterior process, conditioned on evidence. All evidence (both past and future) affect the probability of a specific thinning configuration.

### 4.3 Overview of Our Method

To overcome both of these problems, we extend thinning to an auxiliary Gibbs sampler in the same way that [Rao and Teh, 2011, Rao and Teh, 2013] extended Markovian-model uniformization [Grassmann, 1977] (a specific example of thinning in a Markov process) to a Gibbs sampler. To do this we introduce auxiliary variables representing the events that were dropped. We call these events *virtual* events.

As a standard Gibbs sampler, our method cycles through each variable in turn. In our case, a variable corresponds to an event label. For event label  $l$ , let  $x_l$  be the sampled event sequence for this label. Let  $Y_l$  be all evidence (for  $l$  and other labels) and all (currently fixed) samples for other labels. Our goal is to sample from  $p(x_l | Y_l)$ .

Let  $v_l$  be the virtual events (the auxiliary variable) associated with  $l$  and  $z_l = x_l \cup v_l$  (all event times, virtual and non-virtual). Our method first samples from  $p(v_l | x_l, Y_l)$  and then samples from  $p(x_l | z_l, Y_l)$ . The first step adds vir-

tual events given the non-virtual events are “correct.” The second step treats all events as potential events and drops or keeps events. The dropped events are removed completely. The kept events,  $x_l$ , remain as the new sampled trajectory.

The proof of correctness follows analogously to that of [Rao and Teh, 2013] for Markovian systems. However, the details for sampling from  $p(v_l | x_l, Y_l)$  and  $p(x_l | z_l, Y_l)$  differ. We describe them next.

#### 4.4 Sampling Auxiliary Virtual Events with Adaptive Rates

Sampling from  $p(v_l | x_l, Y_l)$  amounts to adding just the virtual (dropped) events. As the full trajectory ( $x_l$  for all  $l$ ) is known, the rate at any time step for a virtual event is independent of any other virtual events. Therefore, the process is an inhomogeneous Poisson process for which the rate at  $t$  is equal to  $\lambda^* - \lambda_l(t|h)$  where  $h$  is fully determined by  $x_l$  and  $Y_l$ . Recall that  $\lambda_l(t|h)$  is piecewise-constant in time, so sampling from such an inhomogeneous Poisson process is simple.

The auxiliary rate,  $\lambda^*$ , must be strictly greater than the maximum rate possible for irreducibility. We use an auxiliary rate of  $\lambda^* = 2 \max(\lambda(t|h))$  to sample virtual events in the unobserved intervals. This choice balances mixing time (better with higher  $\lambda^*$ ) and computational complexity (better with lower  $\lambda^*$ ).

A naïve way to pick  $\lambda^*$  is to find  $\lambda_{max}$ : the maximum rate in the leaves of PCIM, and use  $2\lambda_{max}$ . However, there could be unobserved time intervals with a possible maximum rate much smaller than  $\lambda_{max}$ . Using  $\lambda_{max}$  in those regions would generate too many virtual events, most of which will be dropped in the next step leading to computational inefficiency. We therefore use an adaptive strategy.

Our adaptive  $\lambda^*(t|h)$  cannot depend on  $x_l$  (this would break the simplicity of sampling mentioned above). Therefore, we determine  $\lambda^*(t|h)$  by passing  $(t, h)$  down the PCIM tree for  $\lambda_l$ . At each internal node, if the branch does not depend on  $x_l$ , we can directly take one branch. Otherwise, the test is related to the sampled events, and we take the maximum rate of taking both branches. This method results in  $\lambda^*(t|h)$  as a piecewise-constant function of time (for the same reasons that  $\lambda_l(t|h)$  is piecewise-constant).

Consider Fig. 3 as an example. When sampling event  $l = A$  on the interval  $[1, 5)$ , we would not take the left branch at the root (no matter what events for  $A$  have been sampled), but must maximize over the other two leaves (as different  $x_l$  values would result in different leaves). This results in a  $\lambda^* = 4$  over this interval, which is smaller than 6.

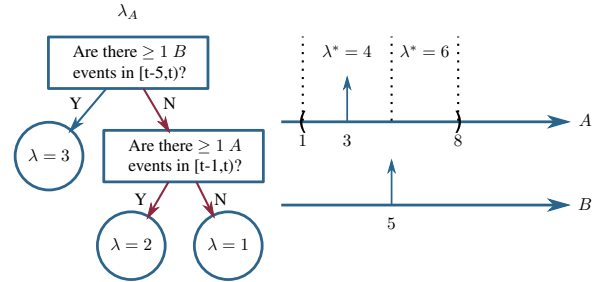


Figure 3: Adaptive auxiliary rate example. When sampling  $A$ , the branch to take at the root does not depend on unobserved events for  $A$ . If the test is related to the sampled event, we take the maximum rate from both branches. The red arrows indicate the branches to take between time  $[1, 5]$ , and  $\lambda^* = 2 \times 2$  in that interval, instead of 6.

#### 4.5 The Naïve FFBS Algorithm

Once these virtual events are added back in, we take  $z_l$  (the union of virtual events and “real” sampled events) as a sample from the Poisson process with rate  $\lambda^*$  and ignore which were originally virtual and which were originally “real.” We then thin this set to get a sample from the conditional marginal over  $l$ .

The restriction to consider events only at times in  $z_l$  transforms the continuous-time problem into a discrete one. Given  $z_l$  with  $m$  possible event times  $(z_{l,1}, z_{l,2}, \dots, z_{l,m})$ , let  $b = \{b_i\}_{i=1}^m$  be a set of binary variables, one per event, where  $b_i = 1$  if event  $i$  is included in  $x_l$  (otherwise  $b_i = 0$  and the event is not included in  $x_l$ ). Thus sampling  $b$  is equivalent to sampling  $x_l$  ( $z_l$  is known) as it specifies which events in  $z_l$  are in  $x_l$ . Let  $Y_l^{i:j}$  be the portion of  $Y$  between times  $z_{l,i}$  and  $z_{l,j}$ , and  $b^{i:j} = \{b_k | i \leq k \leq j\}$ . We wish to sample  $b$  (and thereby  $x_l$ ) from  $p(b | Y) \propto$

$$\left( \prod_i p(Y_l^{i-1:i}, b_i | b^{1:i-1}, Y_l^{1:i-1}) \right) p(Y_l^{m:\infty} | b) \quad (3)$$

where the final  $Y_l^{m:\infty}$  signifies all of the evidence after the last virtual event time  $z_{l,m}$  and can be handled similarly to the other terms.

The most straight-forward method for such sampling considers each possible assignment to  $b$  (of which there are  $2^m$ ). For each interval, we multiply terms from Eq. 3 of the form  $p(Y_l^{i-1:i}, b_i | b^{1:i-1}, Y_l^{1:i-1}) =$

$$p(Y_l^{i-1:i} | b^{1:i-1}, Y_l^{1:i-1}) p(b_i | b^{1:i-1}, Y_l^{1:i-1}) \quad (4)$$

where the first term is the likelihood of the trajectory interval from  $z_{l,i-1}$  to  $z_{l,i}$  and the second term is the probability of the event being thinned, given the past history. The first can be computed by tallying the sufficient statistics (counts and durations) and applying Eq. 2. Note that these sufficient statistics take into account  $b^{1:i-1}$  which specifies

events for  $l$  during the unobserved region(s), and the likelihood must also be calculated for labels  $l' \neq l$  for which  $\lambda_{l'}(t|h)$  depends on events from  $l$ . The second term is equal to  $\frac{\lambda_l(t|h)}{\lambda^*(t)}$  if  $b_i = 1$  (and  $1 - \frac{\lambda_l(t|h)}{\lambda^*(t)}$  if  $b_i = 0$ ). The numerator's dependence on the full history similarly dictates a dependence on  $b^{1:i-1}$ .

This might be formulated as a naïve FFBS algorithm: To generate one sample, we propagate possible trajectories forward in time, multiplying in Eq. 4 at each inter-event interval to account for the evidence. Every time we see a virtual event, each possible trajectory diverges into two (depending on whether the virtual event is to be thinned or not). By the end, we have all  $2^m$  possible trajectories, each with its probability (Eq. 3). We sample one trajectory as the output, in proportion of the calculated likelihoods. As we explicitly keep all possible trajectories, the sampled trajectory immediately tells us which virtual events are kept, so no actual backward pass is needed.

#### 4.6 An Efficient State-Vector Representation

The naïve FFBS algorithm is clearly not practical, as the number of possible trajectories grows exponentially with the number of auxiliary virtual events ( $m$ ). We propose a more efficient state-vector representation to only keep the necessary information for each possible trajectory. The idea takes advantage of the structure of the PCIM and leads to state merges, similar to what happens in FFBS for hidden Markov models (HMMs).

The terms in Eq. 4 depend on  $b^{1:i-1}$  only through the tests in the internal nodes of the PCIM trees. Therefore, we do not have to keep track of all of  $b^{1:i-1}$  to calculate these likelihoods, but only the current state of such tests that depend on events with label  $l$ . For example, a test that asks “Is the last event of label  $l$ ?” only needs to maintain a bit as the indicator. The test “Are there more than 3 events of label  $q$  in the last 5 seconds?” for  $q \neq l$  has no state, as  $b^{1:i-1}$  does not affect its choice. By contrast, a test such as “Is the last event of label  $q$ ?” does depend on  $b$ , even if  $q \neq l$ .

As we propagate forward, we merge  $b^{1:i}$  sequences that result in the same set of states for all internal tests inside the PCIM. See Fig. 4 as a simple example. Though there are 8 possible trajectories, they merge to only 2 states that we can sample from. Similar to FFBS for HMM, we need to maintain the transition probabilities in the forward pass and use them in a backward sampling pass to recover the full trajectory, but such information is also linear.

Note that this conversion to a Markov system for sampling is *not* possible in the original continuous-time system. Thinning allows it by randomly selecting a few discrete time points, thereby restricting the possible state space to be finite.

The state space depends on the actual tests in the PCIM

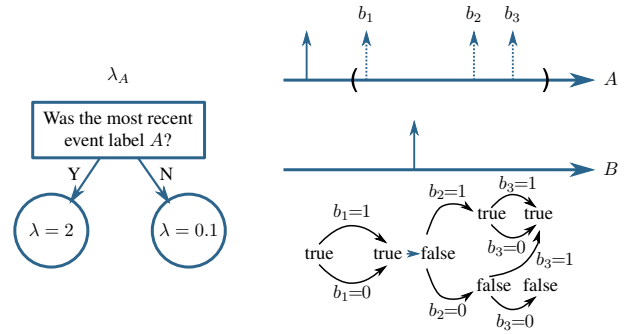


Figure 4: Dotted events are the virtual events that we sample as binary variables ( $b_i$  is 1 if event  $i$  is kept). The state diagram below the trajectory indicates the state of the test as we diverge (keep or drop a virtual event). Though there are  $2^3$  possible configurations, state merges can reduce the exponentially increasing complexity to linear in this case.

model. See Tbl. 1 for the tests we currently support and their state representations. The LastStateTest and StateTest are used to support discrete finite variable systems such as CTBN, as we will use in Sec. 5 and in experiments. Note the EventCountTest was the only supported test in the original PCIM paper. We can see that for tests that only depend on the *current* time (i.e. TimeTest), the diverging history does not affect them, so no state is needed. For Markovian tests (LastEventTest and LastStateTest), we only need a Boolean variable. For the non-Markovian test (EventCountTest), the number of possible states does grow exponentially with the number of virtual events maintained in the queue. This is the best we can do and still be exact. It is much better than growing with the number of all virtual events. However, note that commonly  $lag2 = 0$  and  $n$  is a small number. In this case, the state space size at any point is bounded as  $\binom{m'}{n}$ , where  $m'$  is the maximum number of sampled events in any time interval of duration  $lag1$  (which is upper bounded by  $m$ ). If  $n$  is 1, this is linear in the number of samples generated in during  $lag1$  time units.

As noted above, if the test is not related to the sampled event (for example, in sampling event  $l = A$  with test “are there  $\geq 3$  B events in the last 5 seconds”), the state of the test is null. This is because the evidence and sampled values for  $B$  (not the current variable for Gibbs sampling) can answer this test without reference to samples of  $l$ .

See Alg. 1 for the algorithm description for resampling event  $l$ . The complete algorithm iterates this procedure for each event label to get a new sample. The helper function UpdateState(s,b,t) returns the new state given the old state (s), the new time (t), and whether an event occurs at t (b). SampProbMap(M) takes a mapping from objects to positive values (M) and randomly returns one of the objects with probability proportional to the associate value. AddtoProbMap(M,o,p) checks to see if o is in M. If so, it adds p to the associated probability. Otherwise, it adds the



Table 1: Tests and their corresponding state representations.

| Test           | Example  | State Representation   | Property           |
|----------------|--|--|--------------------|
| TimeTest       | Is the time between 6am and 9am?                       | Null   | independent of $b$ |
| LastEventTest  | Is the last event A?                                   | Boolean  | Markovian          |
| EventCountTest | Are there $\geq n$ A event in $[t - lag1, t - lag2]$ ? | A queue maintaining all the times of A between $[t - lag2, t]$ , and the most recent $n$ events between $[t - lag1, t - lag2]$ . | Non-Markovian      |
| LastStateTest  | Is the last sublabel of var $A=0$ ?                    | Boolean  | Markovian          |
| StateTest      | Is the current sublabel of var $A=0$ ?                 | Null   | independent of $b$ |

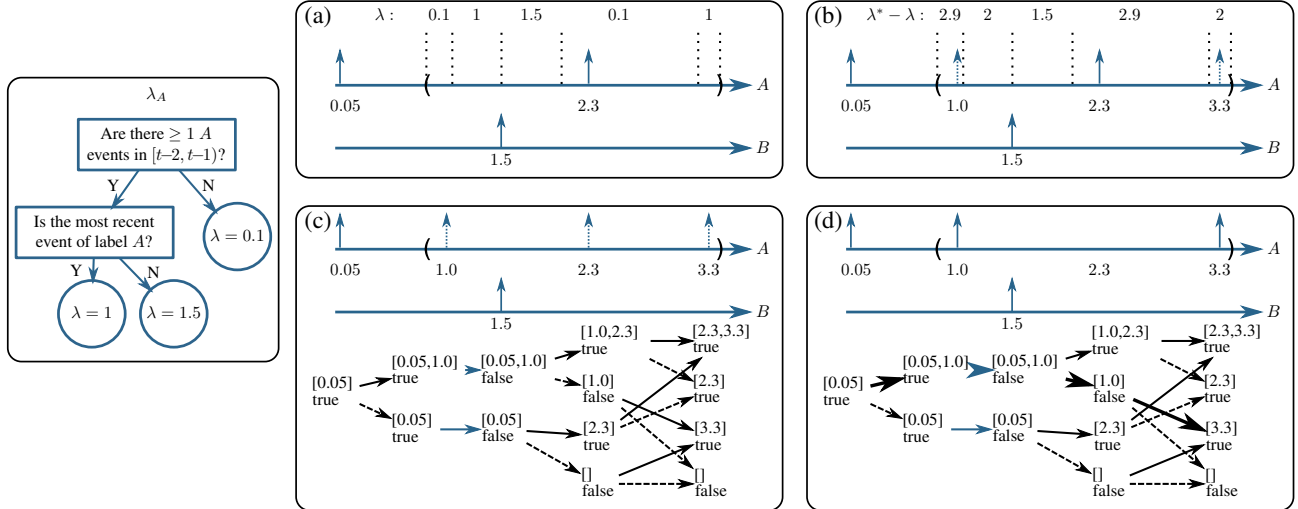


Figure 5: Extended Example, see Section 4.7

mapping  $o \rightarrow p$  to  $M$ .

#### 4.7 Extended Example

Fig. 5 shows an example of resampling the events for label  $A$  on the unobserved interval  $[0.8, 3.5)$ . On the far left is the PCIM rate tree for event  $A$ . Box (a) shows the sample from previous iteration (single event at 2.3). Dashed lines and  $\lambda$  show the piecewise-constant intensity function given the sample. Box (b) shows the sampling of virtual events. For this case  $\lambda^* = 3$  for all time.  $\lambda^* - \lambda$  is the rate for virtual events. The algorithm samples from this process, resulting in two virtual events (dashed). In box (c) all events become potential events. The state of the root test is a queue of recent events. The state of the other test is Boolean (whether  $A$  is more recent). On the bottom is the lattice of joint states over time. Solid arrows indicate  $b_i = 1$  (the event is kept). Dash arrows indicate  $b_i = 0$  (the event is dropped). Each arrow's weight is as per Eq. 4. The probability of a node is the sum over all paths to the node of the product of the weights on the path (calculated by dynamic programming). In box (d) a single path is sampled with backward sampling, shown in bold. This path corresponds to keeping the first and last virtual events and dropping the middle one.

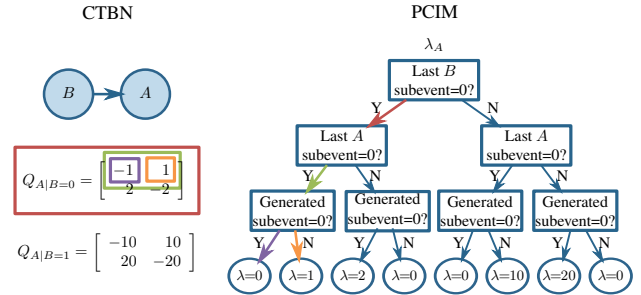


Figure 6: Explicit conversion from a CTBN to a PCIM by using specific tests. Only rates for variable  $A$  shown. The colored arrows and boxes show one-to-one correspondence of a path in the tree and an entry in the rate matrix of CTBN. Diagonal elements in the CTBN are redundant and do not need to be represented in the PCIM.

## 5 REPRESENTING CTBNS AS PCIMS

A non-Markovian PCIM is more general than the Markovian CTBN model. We can, therefore represent a CTBN using a PCIM. In this way, we can extend PCIMs and we can compare our PCIM method with existing methods for CTBNS.

---

**Algorithm 1** Resampling event  $l$ 

---

**input:** The previous trajectory  $(x_l, Y_l)$ **output:** The newly sampled  $x_l'$ 

- 1: **for** each unobserved interval for  $l$  **do**
  - 2: Find piecewise constant  $\lambda^*(t|h)$  using  $Y_l$
  - 3: Find piecewise constant  $\lambda(t|h)$  using  $x_l, Y_l$
  - 4: Sample virtual events  $v_l$  with rate  $\lambda^*(t|h) - \lambda(t|h)$
  - 5: Let  $z_l = x_l \cup v_l$ ,  $m = |z_l|$ , and  $s_0$  be the initial state
  - 6: AddtoProbMap( $S_0, s_0, 1.0$ )
  - 7: **for**  $i \leftarrow 1$  to  $m$  **do**
  - 8: **for** each  $\{(s_{i-1}, \cdot) \rightarrow p\}$  in  $S_{i-1}$  **do**
  - 9:  $p_{keep} = p(E_{i-1:i}, b_i = 1 \mid s_{i-1}, E_{1:i-1})$
  - 10:  $p_{drop} = p(E_{i-1:i}, b_i = 0 \mid s_{i-1}, E_{1:i-1})$
  - 11:  $s_i^{keep} \leftarrow \text{UpdateState}(s_{i-1}, \text{true}, z_{l,i})$
  - 12:  $s_i^{drop} \leftarrow \text{UpdateState}(s_{i-1}, \text{false}, z_{l,i})$
  - 13: AddtoProbMap( $S_i, (s_i^{keep}, z_{l,i}), p \times p_{keep}$ )
  - 14: AddtoProbMap( $S_i, (s_i^{drop}, \emptyset), p \times p_{drop}$ )
  - 15: AddtoProbMap( $T_i(s_i^{keep}), (s_{i-1}, z_{l,i}), p \times p_{keep}$ )
  - 16: AddtoProbMap( $T_i(s_i^{drop}), (s_{i-1}, \emptyset), p \times p_{drop}$ )
  - 17: Update  $S_m$  by propagating until ending time
  - 18:  $x_l' \leftarrow \emptyset$  and  $(s_m', t) \leftarrow \text{SampProbMap}(S_m)$
  - 19: **if**  $t \neq \emptyset$  **then**  $x_l' \leftarrow x_l' \cup \{t\}$
  - 20: **for**  $i \leftarrow m - 1$  to 1 **do**
  - 21:  $(s_i', t) \leftarrow \text{SampProbMap}(T_{i+1}(s_{i+1}'))$
  - 22: **if**  $t \neq \emptyset$  **then**  $x_l' \leftarrow x_l' \cup \{t\}$
  - 23: **return**  $x_l'$
- 

We associate a PCIM label with each CTBN variable. We also augment the notion of a PCIM label to include a sub-label. For each CTBN variable, its PCIM label has one sublabel for each state of the CTBN variable. Therefore, a PCIM event with label  $X$  and sublabel  $x$  corresponds to a transition of the CTBN variable  $X$  from its previous value to the value  $x$ . The PCIM trees' tests can also check the sublabel associated with the possible event.

We augment the auxiliary Gibbs sampler to not only sample which virtual events are kept, but also which sublabel is associated with each. This involves modifying the  $b_i$  variables from the previous section to be multi-valued. Otherwise, the algorithm proceeds the same way.

The last two tests in Tbl. 1 are explicitly for this type of sublabelled event model. We can use them to turn a conditional intensity matrix from the CTBN into a PCIM tree. Fig. 6 shows the conversion of the "twonode" model.

## 6 EXPERIMENTS

We implement our method as part of an open source code base, and all the code and data will be publicly available.

We perform two sets of experiments to validate our method.

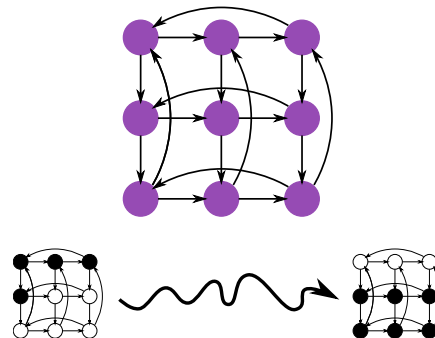


Figure 7: The toroid network and observed patterns [El-Hay et al., 2010].

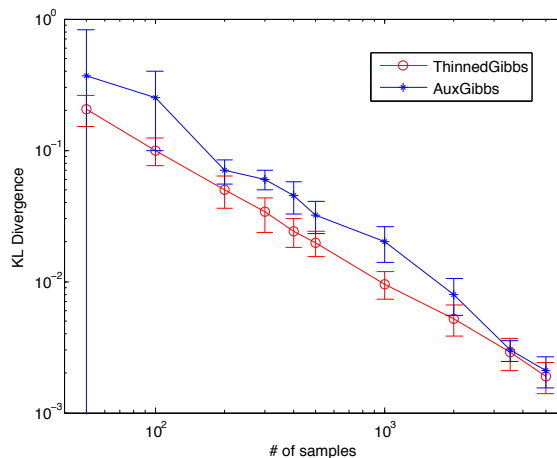


Figure 8: Number of samples versus KL divergence for the toroid network. Both axes are on a log scale.

First we perform inference with our method on both Markovian and non-Markovian models, and compare the result with the ground-truth statistics. For both we show our result converges to the correct result. Ours is the first that can successfully perform inference tasks on non-Markovian PCIMs. For the second set of experiments, we use ThinnedGibbs in EM for both parameter estimation and structural learning for a non-Markovian PCIM. Our inference algorithm can indeed help producing models that achieve higher data likelihood on holdout test data than several baseline methods.

### 6.1 Verification on the Ising Model

We first evaluate our method, ThinnedGibbs, on a network with Ising model dynamics. The Ising model is a well-known interaction model with applications in many fields including statistical mechanics, genetics, and neuroscience. This is a Markovian model and has been tested by several existing inference methods designed for CTBNs.

Using this model, we generate a directed toroid network structure with cycles following [El-Hay et al., 2010].

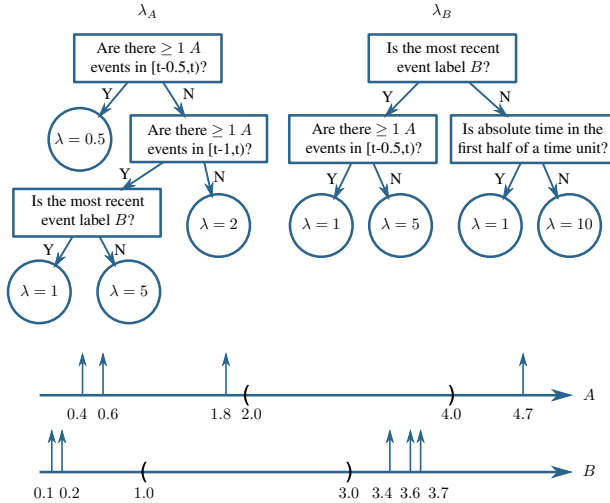


Figure 9: Non-Markovian PCIM and evidence. The ending time is 5.

Nodes can take values  $-1$  and  $1$ , and follow their parents' states according to a coupling strength parameter ( $\beta$ ). A rate parameter ( $\tau$ ) determines how fast nodes toggle between states. We test with  $\beta = 0.5$  and  $\tau = 2$ . The network and the evidence patterns are shown in Fig. 7. The nodes are not observed between  $t = 0$  and  $t = 1$ . We query the marginal distribution of nodes at  $t = 0.5$  and measure the sum of the KL-divergences of all marginals against the ground truth. We compare with the state-of-the-art CTBN Auxiliary Gibbs method [Rao and Teh, 2013]. Other existing methods either produce similar or worse results [Celikkaya and Shelton, 2014]. We vary the sample size between 50 and 5000, and set the burn-in period to be 10% of this value. We run the experiments for 100 times, and plot the means and standard deviations.

Results in Fig. 8 verify that our inference method indeed produces results that converge to the true distribution. Our method reduces to that of [Rao and Teh, 2013] in this Markovian model. Differences between the two lines are due to slightly different initializations of the Gibbs Markov chain and not significant.

## 6.2 Verification on a Non-Markovian Model

We further verify our method on a much more challenging non-Markovian PCIM (Fig. 9). This model contains several non-Markovian EventCountTests. We have observations for event  $A$  at  $t = 0.4, 0.6, 1.8, 4.7$  and for event  $B$  at  $t = 0.1, 0.2, 3.4, 3.6, 3.7$ . Event  $A$  is not observed on  $[2.0, 4.0)$  and event  $B$  is not observed on  $[1.0, 3.0)$ .

In produce ground truth, we discretized time and converted the system to a Markovian system. Note that because the time since the last  $A$  event is part of the state, as the discretization becomes finer, the state space increases. For

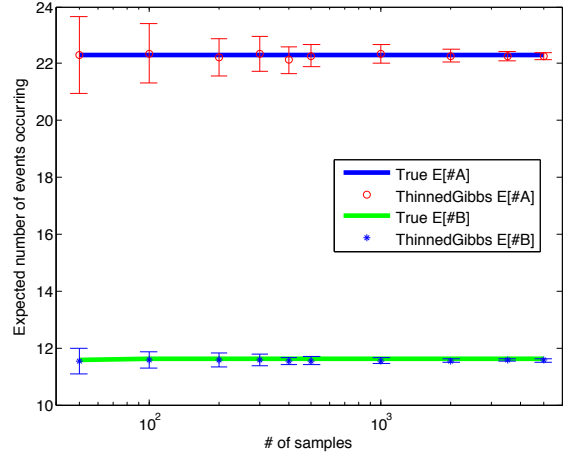


Figure 10: Number of samples versus the inferred expected number of events. The horizontal axis is on a log scale.

this small example, this approach is just barely feasible. We continued to refine the discretization until the answer stabilized. The ground-truth expected total number of  $A$  events between  $[0, 5]$  is 22.3206 and the expected total number of  $B$  events is 11.6161. That is, there are about 18.32  $A$  events and 6.62  $B$  events in the unobserved areas. Note that if the evidence is changed to have no events these numbers drop to 1.6089 and 8.6866 respectively and if the evidence after the unobserved intervals is ignored the expectations are 22.7183 and 8.6344 respectively. Therefore the evidence (both before and after the unobserved intervals) is important to incorporate in inference.

We compare our inference method to the exact values, again varying the sample size between 50 and 5000 and setting the burn-in period to be 10% of this value. We ran the experiments 100 times and report the mean and standard deviation of the two expectations. Our sampler has very small bias and therefore the average values match the true value almost exactly. The variance decreases as expected, demonstrating the consistent nature of our method. See Fig. 10. We are not aware of existing methods that can perform inference on this type of model to which we could compare.

## 6.3 Parameter Estimation and Structural Learning

We further test ThinnedGibbs by using it in EM, for both parameter estimation (given the tree structure, estimate the rates in the leaves), and structural learning (learn both the structure and rates). We use Monte Carlo EM that iterates between two steps: First, given a model we generate samples conditioned on evidence with ThinnedGibbs. Second, given the samples, we treat them as complete trajectories and perform parameter estimation and structural learning, which is efficient for PCIM. We initialize the model from

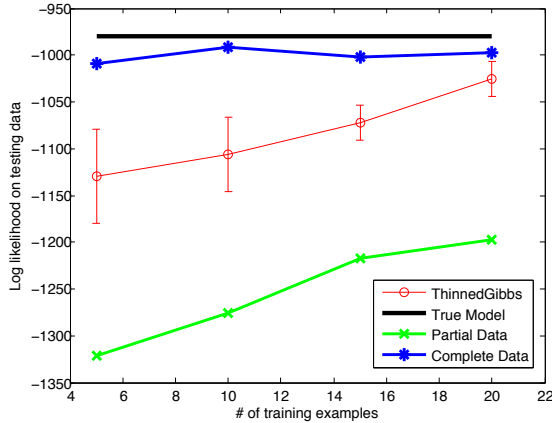


Figure 11: Parameter estimation. Testing log-likelihood as a function of the number of training samples.

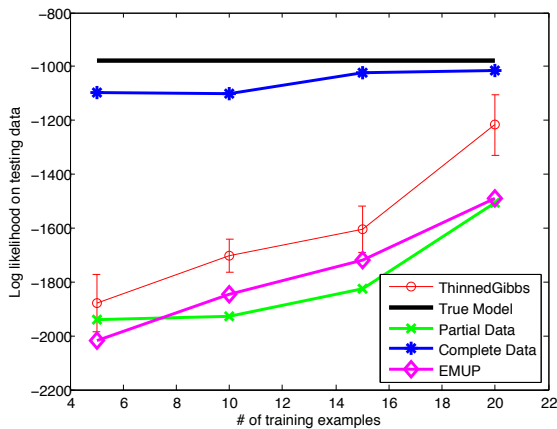


Figure 12: Structure and parameter estimation. Testing log-likelihood as a function of the number of training examples.

the partial trajectories, assuming no events occur in the unobserved intervals. EM terminates when the parameters of PCIMs in two consecutive iterations are stable (all rates change less than 10% from the previous ones), or the number of iterations surpasses 10. For structural learning, the structure needs to be the same between iterations.

We use the model in Fig. 1 and generate complete trajectories for time range  $[0, 10]$ . We vary the number of training samples (5, 10, 15, and 20) and use a fixed set of 100 trajectories as the testing data. For each training size, we use the same the training data for all algorithms and runs. We randomly generate an unobserved interval with length  $0.6 \times T$  for both event labels. For each training sample, ThinnedGibbs fills it in to generate a new sample after burning in 10 steps. For each configuration, we run ThinnedGibbs for 5 times. We measure the data likelihood of the holdout testing data on the learned models.

For parameter estimation, we compare with the true model that generated the data, the model learned with only par-

tial data in which we assume no events happened during unseen intervals (Partial Data), and a model learned with complete training data (Complete Data). The results are summarized in Fig. 11. We can see that the model learned by EM algorithm using ThinnedGibbs can indeed produce significantly higher testing likelihood than using only partial data. Of course, we do not do as well as if none of the data had been hidden (Complete Data).

If learning the structure, there is one other possibility: We could use the original fast PCIM learning method, but indicate (by new event labels) when an unobserved interval starts and stops. We augment the bank of possible decisions to include testing if each pseudo-events have occurred most recently. In this way, the PCIM directly models the process that obscures the data. Of course, at test time, branches modeling such unobserved times are not used. Such model should serve as a better baseline than learning from partially observed data, because it can potentially learn unobserved patterns and only use the dependencies in the observed intervals for a better model. We call this model EMUP (explicit modeling of unobserved patterns).

For structure learning, we fix the bank of possible PCIM tests as EventCountTests with  $(l, n, lag1, lag2) \in \{A, B\} \times \{1, 2\} \times \{2, 3, 4, 5, 6\} \times \{0, 1, 2\}$  (omitting tests for which  $lag1 \leq lag2$ ). For EMUP we also allow testing if currently in unobserved interval. The results are summarized in Fig. 12. We can see that EMUP does outperform models using only partial data. However, Structural EM with ThinnedGibbs still performs better. The performance gain is less than that in the parameter estimation task, probably because there are more local optimums for structural EM, especially with fewer training examples.

## 7 DISCUSSION AND FUTURE WORK

We proposed the first effective inference algorithm, ThinnedGibbs, for PCIM. Our auxiliary Gibbs sampling method effectively transforms a continuous-time problem into a discrete one. Our state-vector representation of diverging trajectories takes advantage of state merges and reduces complexity from exponential to linear for most cases. We build the connection between PCIM and CTBN, and show our method generalizes the state-of-art inference method for CTBN models. In experiments we validate our idea on non-Markovian PCIMs, which is the first to do so.

Our method converges to the exact conditional distribution. If the true state of the model grows exponentially, the complexity of ThinnedGibbs follows. We believe this technique could also be applied to other non-Markovian processes. The challenge lies in computing the forward-pass likelihoods when the rate function is not piecewise-constant.

### Acknowledgement

This work was supported by DARPA (FA8750-12-2-0010).

## References

- [Celikkaya and Shelton, 2014] Celikkaya, E. B. and Shelton, C. R. (2014). Deterministic anytime inference for stochastic continuous-time Markov processes. In *ICML*. 8
- [Cohn et al., 2009] Cohn, I., El-Hay, T., Kupferman, R., and Friedman, N. (2009). Mean field variational approximation for continuous-time Bayesian networks. In *UAI*. 2
- [Dean and Kanazawa, 1988] Dean, T. and Kanazawa, K. (1988). Probabilistic temporal reasoning. In *AAAI*. 1
- [Du et al., 2013] Du, N., Song, L., Gomez-Rodriguez, M., and Zha, H. (2013). Scalable influence estimation in continuous-time diffusion networks. In *NIPS*. 2
- [El-Hay et al., 2010] El-Hay, T., Cohn, I., Friedman, N., and Kupferman, R. (2010). Continuous-time belief propagation. In *ICML*. 2, 7
- [Fan et al., 2010] Fan, Y., Xu, J., and Shelton, C. R. (2010). Importance sampling for continuous time Bayesian networks. *Journal of Machine Learning Research*, 11(Aug):2115–2140. 2
- [Golightly and Wilkinson, 2011] Golightly, A. and Wilkinson, D. J. (2011). Bayesian parameter inference for stochastic biochemical network models using particle Markov chain Monte Carlo. *Interface Focus*. 2
- [Grassmann, 1977] Grassmann, W. K. (1977). Transient solutions in Markovian queueing systems. *Computers & Operations Research*, 4(1):47–53. 2, 3
- [Gunawardana et al., 2011] Gunawardana, A., Meek, C., and Xu, P. (2011). A model for temporal dependencies in event streams. In *NIPS*. 1, 2, 3
- [Lewis and Shedler, 1979] Lewis, P. A. W. and Shedler, G. S. (1979). Simulation of nonhomogeneous Poisson processes by thinning. *Naval Research Logistics Quarterly*, 26(3):403–413. 1, 2, 3
- [Linderman and Adams, 2014] Linderman, S. W. and Adams, R. P. (2014). Discovering latent network structure in point process data. In *ICML*. 2
- [Nodelman et al., 2002] Nodelman, U., Shelton, C. R., and Koller, D. (2002). Continuous time Bayesian networks. In *UAI*. 2
- [Parikh et al., 2012] Parikh, A., Gunawardana, A., and Meek, C. (2012). Cojoint modeling of temporal dependencies in event streams. In *UAI Workshops*. 2
- [Rajaram et al., 2005] Rajaram, S., Graeol, T., and Herbrich, R. (2005). Poisson-networks: A model for structured point process. In *AISStats*. 2
- [Rao and Teh, 2011] Rao, V. and Teh, Y. W. (2011). Fast MCMC sampling for Markov jump processes and continuous time Bayesian networks. In *UAI*. 2, 3
- [Rao and Teh, 2013] Rao, V. and Teh, Y. W. (2013). Fast MCMC sampling for Markov jump processes and extensions. *Journal of Machine Learning Research*, 14:3207–3232. arXiv:1208.4818. 2, 3, 4, 8
- [Saito et al., 2009] Saito, K., Kimura, M., Ohara, K., and Motoda, H. (2009). Learning continuous-time information diffusion model for social behavioral data analysis. In *ACML*, pages 322–337. 2
- [Simma and Jordan, 2010] Simma, A. and Jordan, M. (2010). Modeling events with cascades of Poisson processes. In *UAI*. 2
- [Weiss and Page, 2013] Weiss, J. and Page, D. (2013). Forest-based point processes for event prediction from electronic health records. In *ECML-PKDD*. 2

---

# Memory-Efficient Symbolic Online Planning for Factored MDPs

---

**Aswin Raghavan**  
School of EECS  
Oregon State University  
Corvallis, OR, USA  
nadamuna@eecs.orst.edu

**Roni Khardon**  
Department of Computer Science  
Tufts University  
Medford, MA, USA  
roni@cs.tufts.edu

**Prasad Tadepalli**  
School of EECS  
Oregon State University  
Corvallis, OR, USA  
tadepall@eecs.orst.edu

**Alan Fern**  
School of EECS  
Oregon State University  
Corvallis, OR, USA  
afern@eecs.orst.edu

## Abstract

Factored Markov Decision Processes (MDP) are a de facto standard for compactly modeling sequential decision making problems with uncertainty. Offline planning based on symbolic operators exploits the factored structure of MDPs, but is memory intensive. We present new memory-efficient symbolic operators for online planning, prove the soundness of the operators, and show convergence of the corresponding planning algorithms. An experimental evaluation demonstrates superior scalability on benchmark problems.

## 1 INTRODUCTION

The success of online planning in Markov Decision Processes (MDPs) depends crucially on the extent to which the information gathered from search is generalized to unseen states. In the absence of generalization and heuristic guidance, the planner must explore the entire reachable state space. In factored MDPs, the size of the state and action spaces is exponential in the number of state and action variables, causing algorithms that are polynomial in the number of states and/or actions to be impractical. The current state-of-the-art online algorithms based on e.g., Real-Time Dynamic Programming (RTDP) (Barto *et al.*, 1995) and UCT (Kocsis & Szepesvári, 2006), search in terms of atomic or “flat” states. They are unable to take advantage of the factored structure present in the MDP descriptions which allows strong generalization among states (Boutillier *et al.*, 1999).

In contrast, symbolic decision theoretic planners, such as SPUDD (Hoey *et al.*, 1999), do take advantage of the factored structure. These algorithms interleave Dynamic Programming (DP) (Bertsekas & Tsitsiklis, 1996) updates with steps of model minimization (Givan *et al.*, 2003) in a selected representation such as Algebraic Decision Diagrams (ADD) (Bahar *et al.*, 1993). These offline planners some-

times scale to large MDPs, but depend on compactly representing the optimal value function of the entire MDP (Hoey *et al.*, 1999). Due to this requirement, these algorithms exceed practical memory and time limits in many problems of interest.

Symbolic Real-Time Dynamic Programming (sRTDP) (Feng *et al.*, 2002; Feng & Hansen, 2002) aims to combine the benefits of the symbolic methods and online planning by incorporating symbolic state generalization into the computation of the online planner. This effectively imposes state constraints capturing reachability from the current world state into the symbolic computation. However, sRTDP is a general framework, and its performance is sensitive to the definition of generalized states. Existing definitions in prior work lead to algorithms that exceed memory limits in many cases. Despite the aim for generalization, the resulting planner is often inferior to the corresponding algorithms working in the flat state space (e.g. RTDP).

Our main contribution is the introduction of new symbolic generalization operators that guarantee a more moderate use of space and time, while providing non-trivial generalization. Using these operators, we present symbolic online planning algorithms that combine forward search from an initial state with backwards generalized DP updates. The first algorithm, Path Dynamic Programming (PDP) (Section 3.1), samples fixed-length trajectories by acting greedily and refines *one path* in an ADD for each visited state. It uses either an operator based on value invariance (PDP-V) or one based on policy invariance (PDP- $\pi$ ). Both operators yield anytime algorithms that guarantee convergence to the optimal value and action for the current world state, while maintaining bounded growth in the size of the symbolic representation.

In spite of the slow growth of the value function representation, intermediate computations in PDP leading to that representation can potentially exceed memory limits. This motivates our second operator that performs a more careful control of space in its generalization. The resulting planning algorithm, Pruning Path Dynamic Programming (pPDP) (Section 4), applies the pruning procedure of

Raghavan *et al.* (2013) to control the size of intermediate results. The algorithm is convergent and provides generalization only when it does not increase space requirements compared to flat state search. Thus, it is guaranteed not to be worse than flat state space search. It is the first symbolic algorithm to yield a sound generalization while guaranteeing not to use more memory than flat RTDP.

We empirically demonstrate (Section 5) the performance of PDP and pPDP on three benchmark domains from the recent International Probabilistic Planning Competitions (IPPC), where the proposed algorithms scale significantly better than previous results.

## 2 PROBLEM FORMULATION

### 2.1 Algebraic Decision Diagrams (ADD)

An Algebraic Decision Diagram (ADD) (Bahar *et al.*, 1993; Bryant, 1992) represents a real-valued function  $B^n \rightarrow \mathcal{R}$  over  $n$  boolean variables compactly in the form of a rooted Directed Acyclic Graph (DAG), where each interior node has an associated test variable and two outgoing edges labeled by true or false that lead to its children. An example ADD is shown on the right of Figure 1. Every assignment of variables to truth values traces a unique path to a leaf from the root. Each leaf node contains the value of the ADD function for all assignments  $\mathbf{x}$  that reach that node. If  $D$  is the ADD,  $D[\mathbf{x}]$  represents its evaluation on  $\mathbf{x}$ . In the example, the assignment `reboot.c1=0, running.c1=0`, and `running.c1'=0` leads to the value 0.95. A Binary Decision Diagram (BDD) is an ADD with 0/1 leaves.

We assume that the ADD is *ordered* in that there is a fixed total ordering on the variables that all directed paths in the ADD follow. Ordered ADDs have a canonical representation for any function (although their compactness depends on the ordering) and they support polynomial time operations over the functions they represent. The unary “restrict operator” fixes the value of a variable  $x$  to  $x$  or  $\bar{x}$  in ADD  $D$  and returns a new ADD denoted by  $D_{\downarrow x}$ . In general, a binary ADD operation  $C = A \text{ op } B$  gives an ADD such that  $C[\mathbf{x}] = A[\mathbf{x}] \text{ op } B[\mathbf{x}]$  for every  $\mathbf{x}$ . The result  $C$  is computed symbolically and in polynomial time in the size of the ADDs  $A$  and  $B$ . Any binary operation can be used as *op*, for example,  $\{+, -, \times, \div, \max, \min\}$ . Marginalization operations such as  $\max_V D$ ,  $\min_V D$ ,  $\sum_V D$  are defined naturally over all possible restrictions of  $D$  over values of variables in the set  $V$ , e.g.  $\max_x D \equiv \max(D_{\downarrow x}, D_{\downarrow \bar{x}})$ . We also use the operator  $\oplus_C$  for ADDs, where  $A \oplus_C B = (1 - C)A + CB$ , for a binary valued ADD  $C$ . That is, the result takes the values from  $B$  if  $C$  is true and otherwise from  $A$ . This is similar to the ITE(C,B,A) notation in the BDD literature. This operation can cause merging of paths within the ADD due to reduction, e.g. if  $A$  and  $B$  agree on many values.

A partial assignment is a truth assignment to a subset of variables in  $D$ . An assignment is full if it assigns values to all variables. An extension of a partial assignment is a full assignment which is consistent with it. Every path in the ADD from the root to the leaf defines a partial assignment over the internal variables in that path. For example, the path to 0.95 traversed in the example above, defines a partial assignment to three of the variables but leaves other variables, for example `running.c2`, unspecified.

The *path function* for an ADD  $D$  maps a full assignment  $\mathbf{x}$  to the partial assignment defined by the path traced by  $\mathbf{x}$  in  $D$  and is denoted by  $\Phi(D, \mathbf{x})$ . The path function is represented as an ordered BDD that returns 1 for all assignments consistent with the partial assignment and 0 otherwise. For an assignment  $\mathbf{x}$ , ADD  $D$  and  $\phi = \Phi(D, \mathbf{x})$ , let  $\epsilon(\phi)$  denote the set of all extensions of  $\phi$ .

Two additional transformations are useful. The first converts a BDD  $B$  to an ADD  $D$  by mapping the 0-leaf in  $B$  to  $-\infty$ , denoted by  $D = \underline{B}$ . The second, a complementing operation, converts an ADD  $D$  to a BDD  $B$  by mapping the 0-leaf in  $D$  to 1 and all other real-valued leaves to 0, denoted by  $B = \overline{D}$ .

### 2.2 Factored State and Action MDPs

An MDP (Puterman, 2014) is a tuple  $(\mathcal{S}, \mathcal{A}, T, R)$  where  $\mathcal{S}$  is a finite state-space,  $\mathcal{A}$  is a finite action space,  $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$  denotes the transition function  $T(s, a, s') = Pr(s'|s, a)$ ,  $R : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{R}$  denotes the immediate reward of taking action  $a$  in state  $s$ . In this paper, we focus on finite-horizon planning where the goal is to maximize the expected cumulative reward over a specified horizon  $H$ . A non-stationary policy  $\pi = (\pi_1, \dots, \pi_H)$  is a sequence of mappings such that each  $\pi_i : \mathcal{S} \rightarrow \mathcal{A}$  determines the action to take in a state when there are  $i$  steps-to-go. The value function of a policy  $\pi$  with  $i$  steps-to-go is denoted by  $V_i^\pi(s)$ , which gives the expected total reward of following  $\pi$  starting in state  $s$  for  $i$  steps. The value function of the optimal  $i$ -horizon policy is denoted by  $V_i^*(s)$ .

In a factored MDP (Boutilier *et al.*, 1999) the state space  $\mathcal{S}$  and action space  $\mathcal{A}$  are specified by finite sets of state variables  $\mathbf{X} = (X_1, \dots, X_l)$  and action variables  $\mathbf{A} = (A_1, \dots, A_m)$ . We will assume that the variables are discrete and binary so that  $|\mathcal{S}| = 2^l$  and  $|\mathcal{A}| = 2^m$ .

The transition and reward functions are defined in terms of state and action variables using a Dynamic Bayesian Network (DBN), a two-time-step graphical model that captures the variables at time  $t$  that influence the value of each  $X'_i$  at time  $t + 1$  via the conditional probability functions  $P(X'_i | \text{Parents}(X'_i))$ . The reward function is represented as a node at time  $t + 1$ . Following Hoey *et al.* (1999), the functions are represented using ADDs (Bahar *et al.*, 1993) to

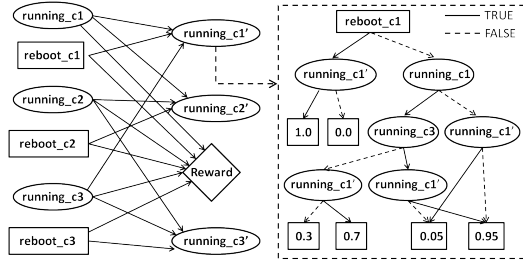


Figure 1: Example of a Factored MDP with Factored Actions. The ADD on the right shows the conditional probability distribution for `running_c1`.

compactly capture sparsity and context-specific dependencies often found in factored MDPs (Boutilier *et al.*, 1999).

For example, Figure 1 shows a DBN for the SysAdmin problem (Guestrin *et al.*, 2001) (see Section 5). The DBN encodes that the computers ‘c1’, ‘c2’ and ‘c3’ are arranged in a directed ring so that the running status of each is influenced by its reboot action and the status of its predecessor. The ADD (right panel) shows that the proposition ‘`running_c1`’ cannot become false if it is ‘rebooted’, and otherwise the next state depends on the status of the neighbors. If currently running, it fails w.p. 0.3 if its neighboring computer ‘c3’ is not operational, and w.p. 0.05 otherwise. A failed computer becomes operational w.p. 0.05.

In previous work we generalized the symbolic approach of SPUDD (Hoey *et al.*, 1999) and introduced algorithms that handle factored states and factored actions. This includes Factored Action Regression (FAR) (Raghavan *et al.*, 2012), a symbolic version of Value Iteration (VI). By leveraging ADD operations (near-)optimal value functions/policies are *deduced* in propositional logic without enumerating the states and actions. Symbolic VI using FAR works by iterating Equation 1,

$$Q = [R + \gamma \sum_{X'_1} P^{X'_1} \times \dots \times \sum_{X'_i} P^{X'_i} \times (V_n)'] \quad (1)$$

with  $V_0 = 0$  and  $V_{n+1} = \max_{A_1 \dots A_m} Q$ . Here  $(V_n)'$  swaps the state variables  $X$  in the diagram  $V_n$  with next state variables  $X'$ , each summation computes the expectation over one  $X'_i$  and marginalizes (and removes)  $X'_i$  from the ADD. Therefore, the ADD  $V_{n+1}$  is the result of one backward DP update for all states.

Below we also use a decision diagram representation of policies. Following (Raghavan *et al.*, 2013) the policy is represented as a BDD over state and action variables and evaluates to 1 on the policy action for a given state. This can be calculated using diagram operations as  $\pi_{n+1} = \overline{\max_A Q - Q}$ , where  $\overline{\phantom{x}}$  is the complementing operator (Section 2.1). Since  $\max_A Q - Q \geq 0$  everywhere and  $> 0$  on paths that include suboptimal actions this identifies the greedy policy with respect to  $Q$ . Below we slightly abuse notation and denote this operation as  $\pi_{n+1} = \arg \max_A Q$ .

Unlike other approaches to factored MDPs based on function approximation (Guestrin *et al.*, 2001; Koller & Parr, 2000), the symbolic algorithms do not require engineered basis functions but rely on compactly representing value functions and policies. FAR and Opportunistic Policy Iteration (Raghavan *et al.*, 2013), a memory-efficient symbolic variant of Modified Policy Iteration (Puterman & Shin, 1978) using FAR, are currently the most effective symbolic algorithms for factored MDPs. As mentioned above, in many cases these methods fail due to the memory exhaustion caused by progressively larger ADDs  $V_n$  as  $n$  increases. Symbolic Online Planning aims to reduce the memory usage by restricting to the state space reachable from the current world state.

### 2.3 Symbolic Online Planning

In order to facilitate the presentation of online symbolic methods we next consider an update that explicitly controls which states are updated. Let  $X$  be a BDD representing some set of states, and let  $\underline{X}$  be the corresponding constraint ADD mapping states in  $X$  to 1, and states not in  $X$  to  $-\infty$ . The operator  $\mathcal{B}^*(V, X)$  performs an exact update (a Bellman backup) for the values of states in  $X$  and copies the values from  $V$  for other states using  $\oplus$ . This operator can be implemented via the ADD expression:

$$\mathcal{B}^*(V, X) \triangleq V \oplus_X [\max_A (R + \gamma E_{X'_1} \dots E_{X'_i} (\underline{X} \times V'))] \quad (2)$$

where multiplication by  $\underline{X}$  is not necessary for correctness but it helps control space. The product of  $V'$  with  $\underline{X}$  fixes the value of states that are not in the set  $X$  to  $-\infty$ . Therefore, the sum and product operations also result in  $-\infty$  without increasing the size of ADDs for these states. The constraint  $\underline{X}$  can be pushed inside the summations due to the distributive property of ADD operations (Raghavan *et al.*, 2013). Note that sRTDP uses an operator equivalent to  $\mathcal{B}^*(V, X)$  via a more memory intensive ADD expression.

We can now explain more general algorithms. Let  $X$  denote a set of states or “a generalized state” and  $s$  denote an atomic state or “flat state”. FAR (Equation 1) uses the backup of Equation 2 with  $X = 1$ , which means it updates the values of all states.

Real-Time Dynamic Programming (RTDP) (Barto *et al.*, 1995) is an online planning algorithm that only updates the values of reachable states. RTDP works by simulating trajectories from a starting state and setting  $X = s$  for each encountered flat state  $s$ . While each update is time and space-efficient, convergence can take a long time in factored MDPs.

sRTDP (Feng *et al.*, 2002) generalizes RTDP updates, uses an update similar to Equation 2 by setting  $X$  to an arbitrary set of states. The set  $X$  is defined by an equivalence relation over states (e.g. the bisimulation relationship (Givan *et al.*, 2003)), which is in practice, heuristically chosen



to trade off the efficiency of the update with the benefit of generalization. An unwise choice of  $X$  in Equation 2 can lead to unreasonable space (or time) requirements. Despite its goal of generalization, the performance of sRTDP can be inferior to RTDP in the online setting where both space and time are limited.

Next we describe our formulations of generalized states  $X$  that lead to efficient updates both in time and space. Convergence of RTDP (and sRTDP) can be retained if  $X$  includes state  $s$ . Efficiency comparable to RTDP can be achieved if the generalized value functions and policies can be captured with about the same amount of memory. We give equivalence relationships that are more restricted than bisimulation (Li *et al.*, 2006), and lead to efficient symbolic online algorithms.

### 3 PDP

Our algorithms are instantiations of Trial-Based Real Time Dynamic Programming (RTDP) (Barto *et al.*, 1995; Keller & Helmert, 2013) with a particular generalized backup function and a fixed trial length. They have two parameters : a lookahead integer  $H > 0$  that is the length of trajectories and a real valued  $\varepsilon$  that controls approximation error in the values of states.

In contrast to most online planners which use a tabular representation, we maintain one value ADD  $V^d$ ,  $d \in [0, H-1]$  per level of the lookahead tree. We chose this over a global ADD (as in sRTDP) because it allows representing non-stationary policies and value functions compactly, as well as allowing different levels of approximation per level, e.g., for increasingly coarse representations of the future. In addition, to simplify the presentation, we explicitly maintain a policy BDD  $\pi^d$ , for each level  $d$ .

Our algorithms start from an initial state and sample a trajectory  $\langle s_0, a_0, \dots, a_{H-2}, s_{H-1} \rangle$  by following the greedy policy  $\pi^0, \dots, \pi^{H-1}$ . Then, the ADDs are updated in the backward direction:  $V^{H-1}$  is updated from the ADD 0,  $V^{H-1}$  is used to update  $V^{H-2}$  and so on till  $V^0$ , which includes  $s_0$  but may be more general.

The general pseudocode for all the algorithms is shown in Figure 2. They have an update of the form  $V = V \oplus_M \max_{\mathbf{A}} Q$ . The algorithm requires three properties: (A)  $M$  is a *path* over state variables (hence the name Path Dynamic Programming), (B)  $Q$  is an ADD over state and action variables with updated values for a super-set of the states in  $M$ . (C) The current state  $s_i$  is included in path  $M$ .

**Proposition 1.** *Any instance of the PDP algorithm (Figure 2) satisfying properties B and C converges to the optimal value (and action) for  $s_0$ .*

**Proof (sketch):** The proof directly follows from the convergence of Trial-Based RTDP (Barto *et al.*, 1995). First,

**Algorithm 3.1:** (ADDs  $V^0, \dots, V^{H-1}, \pi^0, \dots, \pi^{H-1}$ )

```

Initialize each  $V^i \leftarrow (H - i + 1)R_{\max}, \pi^i \leftarrow \text{NoOp}$ .
Sample trajectory  $\langle s_0, a_0, \dots, a_{L-1}, s_L \rangle$  using  $\pi$ .
for  $i \leftarrow L - 1$  downto 0
  if PDP-V then  $(Q, M) \leftarrow$  Equations 4, 5
  if PDP- $\pi$  then  $(Q, M) \leftarrow$  Equations 6, 8
  do  $\left\{ \begin{array}{l} \text{if pPDP then } (Q, M) \leftarrow \text{Equations 9, 11} \\ V^i \leftarrow V^i \oplus_M \max_{\mathbf{A}} Q \\ \pi^i \leftarrow \pi^i \oplus_M \arg \max_{\mathbf{A}} Q \end{array} \right.$ 
  if beyond time or trajectory budget
  then return  $\pi^0(s_0)$ 

```

Figure 2: Pseudocode for Path Dynamic Programming (PDP).  $A \oplus_X B = (1 - X)A + XB$ .

it can be shown that PDP maintains the invariants  $V_i \geq V_i^*$  for all  $i$ . Second, it uses greedy action selection to sample trajectories and each trajectory always includes the state  $s_0$ . Hence if each update is equivalent to DP update on some states, the value and policy at  $s_0$  converge to their optimal values.  $\square$

#### 3.1 PDP-V

The main idea for PDP is to restrict the update to one path in the ADD, instead of one state in RTDP, and PDP-V uses one path in the value ADD. However, this requires a careful control of the set  $M$  as shown below.

$$\mathcal{B}(V, s) = V \oplus_M \max_{\mathbf{A}} Q \quad (3)$$

$$Q = R + \sum_{X'_1} P_1 \times \dots \times \sum_{X'_l} P_l \times (\Phi(V, s) \times V') \quad (4)$$

$$M = \Phi(\max_{\mathbf{A}} Q, s) \wedge \Phi(V, s) \quad (5)$$

For a given state  $s$  and value ADD  $V$ , the values of all states that are extensions of the current path  $\Phi(V, s)$  are updated in the ADD  $Q$  (Equation 4). The ADD  $\max_{\mathbf{A}} Q$  has updated values for the path extensions of  $\Phi(V, s)$  and  $-\infty$  otherwise. But using this update with  $M = \Phi(V, s)$  might lose compactness if many of the extensions of  $\Phi(V, s)$  have different values. Additionally, the new path  $\Phi(\max_{\mathbf{A}} Q, s)$  can be shorter than  $\Phi(V, s)$  whereas only the extensions  $\Phi(V, s)$  have correctly updated values. Sound generalization and compactness are both achieved by restricting the update to the path refinement of  $\Phi(V, s)$  by setting  $M$  to be the intersection of the set of states that share the same path as state  $s$  before and after the update. This is the main difference between PDP and sRTDP. It guarantees that the updated  $V$  has the same number of leaves as the flat state update, an important guarantee for a symbolic method.

**Proposition 2.** *Let  $V$  be an ADD,  $s$  a state,  $W = \mathcal{B}(V, s)$*

and  $M$  the path according to Equation 5.

- (a) For all states  $q \in \epsilon(M)$ ,  $W[q] = \mathcal{B}^*(V, q)[q]$ .  
 (b)  $W$  grows by at most one leaf node over  $V$ .

**Proof (sketch):** (a) follows because ADD  $Q$  is a sound update for states in  $\Phi(V, s)$  (because the constraint  $\Phi(V, s)$  can be pushed inside the summation as in Raghavan *et al.* (2013)). The BDD  $M$  represents a subset of states that satisfy  $\Phi(V, s)$  due to Equation 5<sup>1</sup>. (b) is true because the path  $M$  leads to a leaf in  $\max_{\mathbf{A}} Q$ . For paths in  $1 - M$  the values are copied from  $V$  and do not add leaves to  $W$ .  $\square$

The first part of Proposition 2 guarantees the convergence of PDP-V according to Proposition 1. In practice, it is observed that the paths in symbolic VI often remain unchanged between consecutive iterations while the values have not converged. PDP-V updates these efficiently and succinctly, gaining a speedup proportional to the number of states in the path. However, in order to find the mask  $M$  in PDP-V we have to calculate updated values for all extensions of  $\Phi(V, s)$  in Equation 4, and in some cases this preparatory step exceeds memory limits. Section 4 gives an algorithm that does not have this disadvantage.

### 3.2 PDP- $\pi$

PDP- $\pi$  similarly restricts the update to one path. However, it appeals to the notion of policy irrelevance (Jong, 2005; Li *et al.*, 2006; Hostetler *et al.*, 2014), that captures states having the same optimal action. Recall that PDP maintains a policy representation in addition to the value ADDs. PDP- $\pi$  updates states that share a path in  $\pi$  before and after a DP update to the policy. The memory efficiency of path refinement is retained with respect to the policy representation.

PDP- $\pi$  starts with a trivial policy (e.g. NoOp) and refines the policy for generalized states visited by trajectories. In this way, PDP- $\pi$  behaves more like a policy search method. It is well known that in some cases paths in the policy BDD remain unchanged during iterations of symbolic VI even though the values keep changing. PDP- $\pi$  captures these succinctly (Section 5).

Let  $\pi(s)$  be the policy action, i.e., a complete assignment to action variables for state  $s$  according to BDD  $\pi$ ,  $\pi(s) = \arg \max_{\mathbf{A}} \pi_{\downarrow s}$ . In case of a tie, some action variables are set to false (including the case when they are unspecified by  $\pi_{\downarrow s}$ ). Let  $\Phi_{\pi}(s)$  be the path over state variables according to the greedy action in  $\pi$ ,  $\Phi_{\pi}(s) = \Phi(\pi, s \wedge \pi(s))$ . The update is similar to PDP-V except it uses  $\Phi_{\pi}$  and  $\arg \max$

<sup>1</sup>Note that the proposition does not hold for the path  $\Phi(W, s)$  (rather than  $M$ ) due to the  $\oplus$  operator which might merge an updated path with a path that was not updated.

instead.

$$Q = (R + \sum_{X'_1} P_1 \times \dots \sum_{X'_l} P_l \times (\Phi_{\pi}(s) \times V')) \quad (6)$$

$$\mu = \arg \max_{\mathbf{A}} Q \quad (7)$$

$$M = \Phi_{\mu}(s) \wedge \Phi_{\pi}(s) \quad (8)$$

The ADD  $Q$  contains updated values for the states in  $\Phi_{\pi}(s)$  rather than  $\Phi(V, s)$  as in PDP-V. The mask  $M$  uses  $\mu = \arg \max_{\mathbf{A}} Q$  to denote the greedy policy BDD extracted from  $Q$ . Finally, the policy BDD is updated as  $\pi = \pi \oplus_M \arg \max_{\mathbf{A}} Q$ . Clearly, the property in Proposition 2 (a) holds here as well and therefore by Proposition 1 the algorithm PDP- $\pi$  converges.

## 4 pPDP

The idea in pPDP is to repeatedly prune the intermediate ADDs of Equation 4 so that the ADD  $Q$  has space complexity no larger than the DP update of a flat state. We use the pruning operator proposed in (Raghavan *et al.*, 2013) to control the size of the ADD. Briefly, the pruning operator denoted by  $\mathcal{P}(D, C)$  for an ADD  $D$  and a constraint  $C$  represented as a BDD returns an ADD which is no larger than  $D$ . The result of pruning removes some of the paths from  $D$  that violate the constraint  $C$  but not all.

**Lemma 1.** (Raghavan *et al.*, 2013). *Let  $G = \mathcal{P}(D, \pi)$  then*

- (1) *Every path in  $G$  is a sub-path of a path in  $D$ .*
- (2) *If a path  $p$  in  $G$  does not lead to  $-\infty$ , then for all extensions  $y \in \epsilon(p)$ ,  $G(y) = D(y)$ .*
- (3) *If a path  $p$  in  $G$  does lead to  $-\infty$ , then for all extensions  $y \in \epsilon(p)$  either  $\pi(y) = -\infty$  or  $D(y) = -\infty$ .*

Part (1) gives a strong memory guarantee that  $G$  is no larger than  $D$ . Pruning accomplishes this by leaving some paths in  $G$  unchanged if only some (not all) of their extensions violate the constraint. pPDP uses the flat state  $C = s$  as the constraint. Let  $\mathcal{P}_s(D)$  denote  $\mathcal{P}(D, s)$  for any ADD  $D$  and a flat state  $s$ .

$$Q = \mathcal{P}_s(R + \mathcal{P}_s(\sum_{X'_1} P_1 \times \dots \mathcal{P}_s(\sum_{X'_l} P_l \times V'))) \quad (9)$$

As the expectation is computed over  $X'_i$ , state and action variables are introduced into the paths of  $V'$ . The paths that do not cover the current state are pruned and point to  $-\infty$ . Hence it is efficient to compute the ADD  $Q$  in memory.

**Proposition 3.** (1)  *$Q$  contains  $O(2^m)$  paths over state and action variables that do not lead to  $-\infty$ .*

(2) *Every path  $p$  that does not lead to  $-\infty$  is a DP update for the  $Q$ -value of all states and actions in  $p$ .*

**Proof (Sketch) :** (1) is due to using the state  $s$  as the constraint. Any path that has assignments to state variables

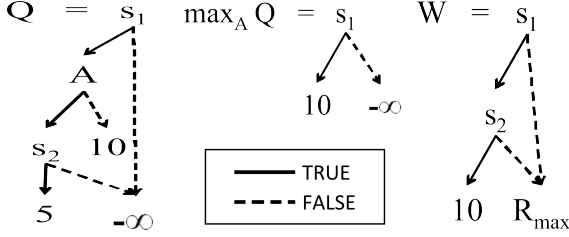


Figure 3: Example illustrating the mask  $M$  in pPDP.  $V$  is set to  $R_{\max}$  initially.  $Q$  is computed using Equation 9 for the state  $s_1 = 1, s_2 = 1$ . The ADD  $\max_{\mathbf{A}} Q$  gives an incorrect value for  $s_1 = 1, s_2 = 0$ , compared to  $W$ , the update using PDP-V (Equation 3).

differing from  $s$  is pruned and leads to  $-\infty$ . The paths that do not lead to  $-\infty$  have different assignments to action variables for a maximum of  $O(2^m)$  paths. (2) is due to part two of Lemma 1 because the pruning operator does not alter the paths that are consistent with state  $s$ .  $\square$

The pruning operator removes portions of the diagram in subtle ways and therefore we have to be careful in choosing the mask  $M$ . Consider using the path  $M = \Phi(\max_{\mathbf{A}} Q, s)$  which at first appears to be a natural choice. Unfortunately, this path does not give sound generalization because of the maximization.

To illustrate this, consider the hypothetical diagram  $Q$  shown on the left of Figure 3 where the state  $s$  assigns  $s_1 = 1$  and  $s_2 = 1$  and paths disagreeing with this assignment have been replaced with  $-\infty$ . The diagram  $\max_{\mathbf{A}} Q$  is shown on the right and gives a value of 10 for state  $s_1 = 1$  and  $s_2 = 0$ . This is incorrect since the true value of  $Q$  from the path  $s_1 = 1, a = 1, s_2 = 0$  can be larger than 10. In  $\max_{\mathbf{A}} Q$  the true value of the states on this path is ignored and assumed to be  $-\infty$ , and therefore the value calculated for the partial assignment  $s_1 = 1$  is not correct for all extensions.

To guarantee correctness we require that all the values in the sub-diagram below the node to be different than  $-\infty$ . Therefore, states whose values are valid in  $\max_{\mathbf{A}} Q$  are those where for all actions  $\mathbf{A}$ , ( $Q \neq -\infty$ ), denoted by the BDD  $\forall_{\mathbf{A}}(Q \neq -\infty)$ . In this example, ( $Q \neq -\infty$ ) when  $\{s_1 = 1, A = 0\} \vee \{s_1 = 1, A = 1, s_2 = 1\}$ , and quantification yields the mask  $M = \{s_1 = 1, s_2 = 1\}$ . Note that the BDDs  $(Q \neq -\infty)$  and  $\forall_{\mathbf{A}}(Q \neq -\infty)$  cannot be zero because all actions are updated in state  $s$ . Therefore, in the worst case, the mask  $M$  is equal to the state  $s$  and the step degenerates to a flat RTDP update. Formally, the operator used in pPDP is

$$\hat{\mathcal{B}}(V, s) = V \oplus_M \mathcal{P}_s(\max_{\mathbf{A}} Q) \quad (10)$$

$$M = \Phi(\forall_{\mathbf{A}}(Q \neq -\infty), s) \quad (11)$$

**Proposition 4.** Given an ADD  $V$  and state  $s$ , let  $W = \hat{\mathcal{B}}(V, s)$  as in Equation 10, and let  $M$  be the path from 11.

Then,  $\forall q \in \epsilon(M), W[q] = \mathcal{B}^*(V, q)[q]$ .

The proof follows from the soundness of pruning (Lemma 1) and the fact that all path extensions of  $M$  lead to a value not equal to  $-\infty$  in  $Q$ . Therefore, by Proposition 1 pPDP converges as well. In cases where PDP exceeds memory limits pPDP can capture some of the sound generalizations, precisely those that can be captured without increasing the size of intermediate  $Q$  ADD. The only overhead in pPDP is the time required for the pruning operations which is negligible.

## 5 EXPERIMENTS

We now evaluate the empirical impact of our proposed generalization operators within the family of RTDP-style algorithms. To do this we compare our algorithms PDP-V, PDP- $\pi$ , and pPDP to the following baselines: 1) **RTDP(Table)** is a simple table-based implementation of RTDP with state values initialized to  $R_{\max}$ . 2) **RTDP(ADD)** is like RTDP(Table) (i.e. no state generalization), except that each state backup is done symbolically using the FAR operator. This can be more efficient for factored actions compared to enumerating actions. 3) **sRTDP** (Feng *et al.*, 2002), where our implementation uses FAR for updates in order to exploit factored actions. 4) **LR<sup>2</sup>TDP** (Kolobov *et al.*, 2012) is an extension of RTDP(Table) to solve finite horizon MDPs using iterative deepening and labeling, which was successful in recent planning competitions. 5) **FAR** (Raghavan *et al.*, 2012) as described above. FAR is limited to 500 minutes of offline planning and then the resulting policy is executed online. This algorithm is only applicable to some of the small problem instances in our experiments and is included to give an optimal baseline value when possible.

All planners were implemented in a common framework, except for LR<sup>2</sup>TDP, for which we used the publicly available code. For PDP-V and pPDP, we initialized each  $V^i$  with  $R_{\max}$  (scaled by  $i$ ). For PDP- $\pi$ , we initialized  $\pi^i$  to the NoOp policy. Planning domains and problems are specified in the Relational Dynamic Influence Diagram Language (RDDI) (Sanner, 2010), which we convert to an ADD-based representation. The ADD variable ordering puts  $\text{parents}(X'_i)$  above  $X'_i$ , where the  $X'_i$ 's are ordered (ascending) by the number of parents that are action variables. Note that the parents include current state variables and action variables so that this defines an ordering over all variables. In all experiments, our symbolic operators allow an approximation error of  $\epsilon = 0.1$  with the upper bound merging strategy (St-Aubin *et al.*, 2001).

Our experiments below are on 5 problem instances of varying sizes from three domains of the 2011 and 2014 International Probabilistic Planning Competitions (IPPCs). A memory limit of 4G is imposed to restrict the size of the

ADDs, beyond which the planner can no longer proceed which we denote as “EML”(Exceeded Memory Limit). A planner is evaluated on a problem by running 30 trials of horizon 40 and averaging the total reward across the trials. We report the averages and 95% confidence intervals for each problem. Planners use a specified time limit per decision and we give results for different time limits. The value functions and policies are reinitialized after each decision.

**Academic Advising Problem:** The Academic Advising domain (Guerin *et al.*, 2012), from IPPC 2014, is a stochastic cost minimization problem that models the process of selecting the courses for a student in order to complete degree requirements, where the courses have complex prerequisite structure. The state space encodes which courses have been taken and whether they were passed or not. The actions at each step correspond to selecting one or more courses to take next. We consider two variants of the domain, a non-concurrent variant, which only allows a single course to be selected at each decision point, and a concurrent version, which allows multiple courses to be selected. The dynamics encode the probability that a course is passed if taken. Missing requirements and retaking of courses are penalized.

Figure 4<sup>2</sup> shows the performance vs. time for the different planners on IPPC 2014 problem instances for the non-concurrent and concurrent variants. All algorithms use a planning lookahead horizon of 16 steps. In the non-concurrent variant and shortest time limit (top left panel), we see that sRTDP fails to scale beyond the two smallest problems, and that FAR is able to solve these two problems as well. In larger instances both of these methods EML. In contrast, PDP-V, PDP- $\pi$  and pPDP are able to give good performance across problem sizes. Moreover, for the smallest instances where FAR is able to compute an optimal policy, these algorithms yield near optimal performance. This result demonstrates the importance of using update operators that attempt to trade-off generalization and memory usage.

The flat search methods RTDP(Table), RTDP(ADD), and LR<sup>2</sup>TDP do not perform well. For the largest three instances, these methods have a return no better than that of a NoOp policy. This shows the importance of generalization across states in order to achieve good performance in reasonable time.

Comparing performance across time limits (increasing time from left to right) we see the following. The flat search methods are not able to improve by much as the time limit is increased. PDP-V, PDP- $\pi$  and pPDP also do not improve significantly with more time. Importantly they are able to avoid EML as more trajectories are sampled with larger time limits. PDP-V shows the most improvement on the largest instance as time increases. This shows that, in this

domain, the use of generalization by our methods is the dominating factor in improving performance, and is even more effective than increasing the time limit.

Figure 4 (bottom) shows results for the same problem instances, but with concurrency (of 5 for the first instance and 2 for others). Here, both sRTDP and FAR (not shown) EML even for the smallest instances. The flat search methods degrade quickly as the problem instances become larger. Our proposed methods (with one exception) outperform the competitors, especially for the larger instances. The exception is PDP- $\pi$  on the largest instance, which results in EML after 18 seconds of planning due to the size of the intermediate ADDs in Equation 6. If we increase the time further (not shown here), PDP-V also does EML. On the other-hand pPDP does not result in EML due to its guarantees on bounding the diagram size (including intermediate diagrams), possibly at the expense of less aggressive generalization.

**SysAdmin Problem :** SysAdmin (Guestrin *et al.*, 2001) models a computer network with  $n$  computers. Computers can fail with some probability, which requires a reboot action to correct. Neighbors of a failed computer have a higher probability of failing. The reward is based on the number of running computers with a cost associated with a reboot action. Unlike the academic advising domain, the number of reachable states in this domain is practically the entire state space. To allow sRTDP to produce non-EML results we consider networks of 10 computers connected in a star network. Following Raghavan *et al.* (2012), we consider problems that vary the maximum number of computers that can be rebooted per decision (1, 3, 5, 7, or 10), which gives a progressively growing factored action space.

Figure 5 gives results for three different time-limit settings. Due to the highly random nature of the problem, we used a short lookahead of four steps for all algorithms. The curve above the bar graphs shows the performance of the optimal policy found by FAR.

sRTDP exhibits interesting behavior in this domain. It performs worse than using no state generalization (i.e. RTDP(ADD)) in the first four instances and then optimally for the largest instance. The increased complexity of the sRTDP backup causes poor performance in the smaller action spaces—sRTDP samples fewer trajectories than our algorithms. On the largest instance as the value ADD becomes more compact (more states have similar values), sRTDP is able to exploit generalization. This shows the difficulty of predicting apriori how much space and time the sRTDP generalization operator may require. In this domain, PDP-V and pPDP scale similarly to RTDP(ADD), because the reward function involves counting the number of computers, which makes the path formula  $\Phi(R, s)$  the same as  $s$ . This means that these algorithms achieve very little state generalization in this domain. However, they

<sup>2</sup>Charts best viewed in color.

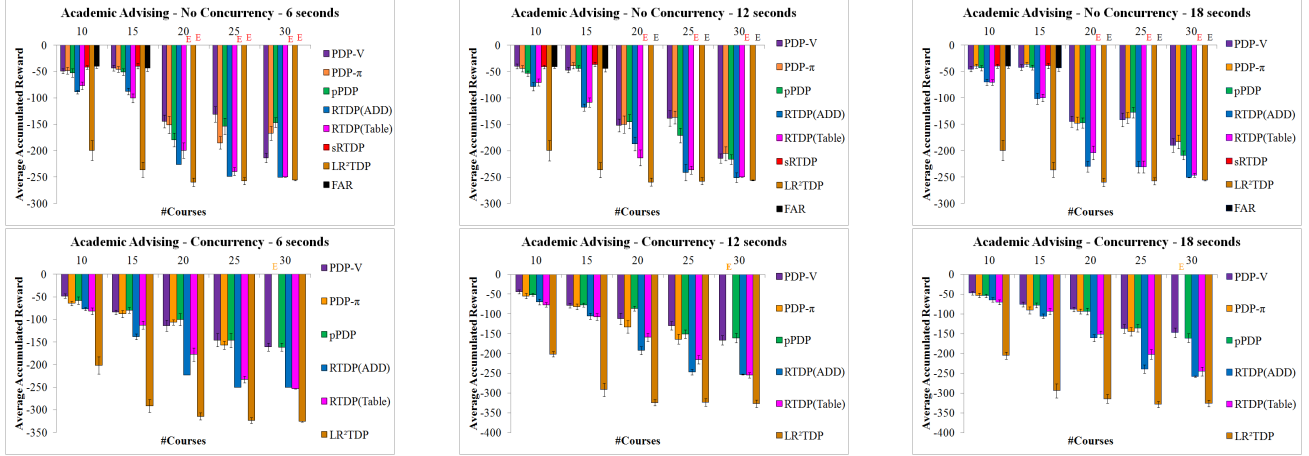


Figure 4: Academic Advising Problem : Performance vs. Time for time limits 6, 12 and 18 seconds per decision without (with) concurrency in the top (bottom) panels respectively. Error bars show 95% confidence intervals. The state space is  $2^{2x}$ ,  $x$  is the number of courses shown on the x-axis.

do outperform RTDP(Table) due to the use of the factored FAR backup compared to a backup based on action enumeration.

In this domain, PDP- $\pi$  clearly outperforms PDP-V and pPDP. In this case, policy irrelevance is able to capture abstract states more succinctly. For example, in the first instance, any state in which the computer at the center of the network is down has the same path formula :  $\sim \text{running}_{c1} \Rightarrow \text{reboot}_{c1}$ . Note that the values of these states are not equal and depend on the status of other computers. As parallelism increases, nodes near the center get added to these rules regardless of the status of nodes farther away. Clearly, in this domain, generalization based on policy irrelevance is more appropriate than value irrelevance.

Finally we see that as the time limit increases there is a small improvement in performance for most algorithms. It is clear that the increase in performance due to larger time limits is not as significant as using the appropriate generalization mechanism, in this case policy irrelevance.

**Crossing Traffic Problem:** This IPPC 2011 domain models the arcade game Frogger, where the agent moves in a 2-D grid to cross a road to reach a goal location, while avoiding right-to-left moving cars that enter the road randomly from the rightmost column. The reward is -1 for each move and -40 for collision.

The boolean encoding of this domain has  $|S| = O(2^{2(n^2)})$  for an  $n \times n$  grid, with two bits per cell for the presence of the agent and car respectively. However, depending on the current location of the agent, many of these bits can be ignored for predicting the optimal value and action.

Figure 6 shows the results for different time limits and problem instances involving 3x3, 4x4, and 5x5 grids. There is larger variance in this domain, compared to the others, due to collisions. We see that sRTDP performs well in the

first three instances and is able to improve with more time per decision. However, in instances 4 and 5 sRTDP exceeds memory limits when given more time. PDP-V and PDP- $\pi$  scale to these instances and times without EML. PDP- $\pi$  performs better than PDP-V initially but PDP-V is able to improve more than PDP- $\pi$  with more time per decision. We see that these algorithms outperform RTDP(ADD), showing that generalization is clearly useful in this domain. Again we see that generalizing appropriately is the dominating factor toward performance compared to increasing the time limit. pPDP never performs worse than the flat search methods RTDP(Table) and RTDP(ADD), and outperforms them in some cases. Its less aggressive generalization, however, leads to overall worse performance compared to our other algorithms.

**Large instances :** The preprocessing of translating RDDDL to propositional logic does not scale for the large instances from the IPPC. For the purpose of showing the scaling of our algorithms, we refactored the RDDDL domain - by decomposing the “robot-at(x,y)” propositions into two independent propositions “robot-at(x)” and “robot-at(y)” because the actions’ effects are independent along  $x$  and  $y$  dimensions. Figure 8 shows the performance of PDP, PDP- $\pi$ , pPDP and sRTDP for grids of sizes  $6 \times 6$  and  $7 \times 7$ .

The algorithms are given much more time per decision to demonstrate the comparative scaling. The charts show the percentage out of 30 trials that the agent reached the goal. We see that in the 6x6 grid (left panel) PDP-V outperforms sRTDP by a large amount. sRTDP is able to scale with time without EML and slowly converges to optimal performance. By comparison, sRTDP does not perform well in the 7 by 7 problem (right panel) whereas PDP-V is able to make progress. PDP- $\pi$  performs worse (better) than sRTDP in the former (latter) instance. The flat search methods are not able to make any progress in this problem due

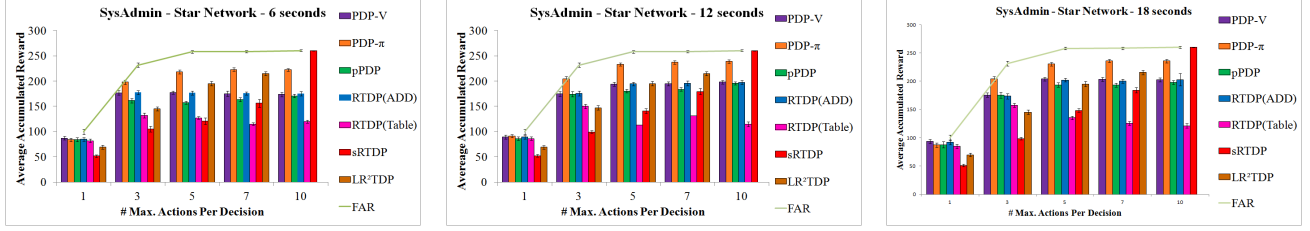


Figure 5: SysAdmin Problem vs. concurrent actions : The state space is  $2^{10}$  and action space is  $O(2^x)$ ,  $x$  is the maximum number of parallel actions.

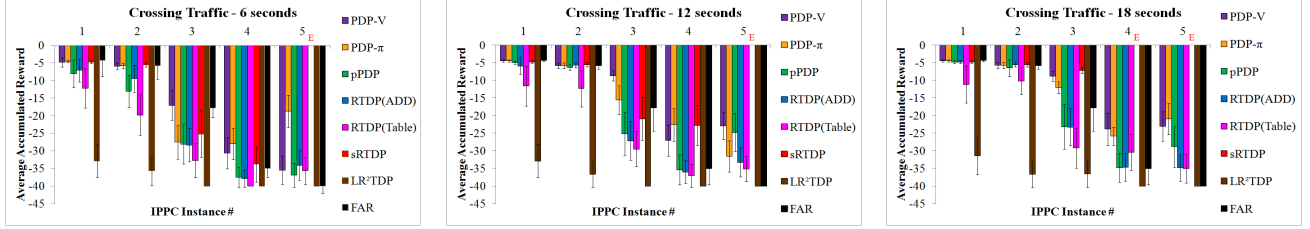


Figure 6: Crossing Traffic Problem : The odd numbered instances have  $3 \times 3$ ,  $4 \times 4$ , and  $5 \times 5$  grids arrival probability of 30%. The even numbered instances have the same grid sizes but with arrival probability of 60%.

Performance vs. Time for 6, 12 and 18 seconds per decision. Error bars show 95% confidence intervals.

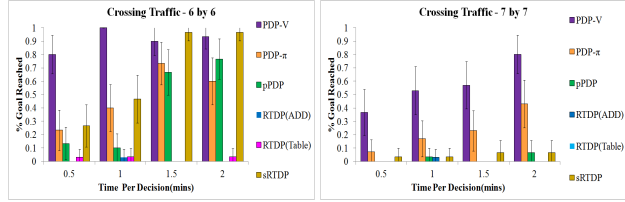


Figure 8: Crossing Traffic (large instances) : Performance vs. Time with 30, 60, 90 and 120 seconds per decision. Error bars show 95% confidence intervals.

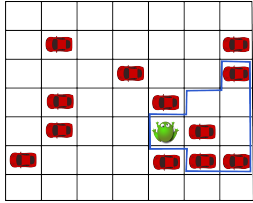


Figure 9: A generalized state discovered by PDP-V in the Crossing Traffic problem. The grid denotes a flat state  $s_0$  and the tiles in blue denote the generalized state  $\Phi(V_0, s_0)$ .

to the large stochastic branching factor. pPDP is able to improve on the flat search in the first instance but falls back to the flat method in the larger problem.

Finally, we present a generalized state found by PDP-V to illustrate the effectiveness of generalization in these problems. Figure 9 shows an instance of the Crossing Traffic problem. All the cells in the grid, including the location of the agent and each car, constitute a flat state. The cells within blue denote the cells that appear in the path formula  $\Phi(V_0, s_0)$  after running PDP-V from  $s_0$ . We see that PDP-

V is able to ignore the assignments to many cells that are irrelevant for optimal online planning.

## 6 SUMMARY

We presented the first fully symbolic planning algorithms for factored MDPs that generalize simulated experience soundly and efficiently. This work is orthogonal to research on improving the anytime performance of RTDP algorithms via smart sampling (McMahan *et al.*, 2005; Walsh *et al.*, 2010) and heuristics (Kolobov *et al.*, 2012; Keller & Helmert, 2013). There were several observations from our experimental results. First, in all domains, we saw that using the appropriate form of generalization was the dominating factor towards good performance compared to increasing the time-limit for algorithms without state generalization. Second, we saw that the most appropriate form of generalization can differ across domains and sometimes problem instances within a domain. This suggests that it is fruitful to investigate mechanisms for tuning or selecting among generalization methods. Third, pPDP never exceeded memory limits, while other generalization approaches did occasionally. Further, previous versions of sRTDP, very frequently exceeded memory limits. This suggests that pPDP is perhaps the safest choice for generalization, especially for large problems and short time limits.

## Acknowledgements

This work was supported by NSF under grants IIS-0964705 and IIS-0964457.

## References

- Bahar, R Iris, Frohm, Erica A, Gaona, Charles M, Hachtel, Gary D, Macii, Enrico, Pardo, Abelardo, & Somenzi, Fabio. 1993. Algebraic Decision Diagrams And Their Applications. *In: Computer-Aided Design*.
- Barto, Andrew G, Bradtke, Steven J, & Singh, Satinder P. 1995. Learning To Act Using Real-Time Dynamic Programming. *Artificial Intelligence*, **72**(1).
- Bertsekas, Dimitri P., & Tsitsiklis, John N. 1996. *Neuro-Dynamic Programming*.
- Boutilier, Craig, Dean, Thomas, & Hanks, Steve. 1999. Decision-Theoretic Planning: Structural Assumptions And Computational Leverage. *Journal Of Artificial Intelligence Research (JAIR)*, **11**(1).
- Bryant, Randal E. 1992. Symbolic Boolean Manipulation With Ordered Binary-Decision Diagrams. *ACM Computing Surveys (CSUR)*, **24**(3).
- Feng, Zhengzhu, & Hansen, Eric A. 2002. Symbolic Heuristic Search for Factored Markov Decision Processes. *In: Eighteenth National Conference on Artificial Intelligence*.
- Feng, Zhengzhu, Hansen, Eric A, & Zilberstein, Shlomo. 2002. Symbolic Generalization For Online Planning. *In: Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence (UAI)*.
- Givan, Robert, Dean, Thomas, & Greig, Matthew. 2003. Equivalence Notions And Model Minimization In Markov Decision Processes. *Artificial Intelligence*, **147**(1).
- Guerin, Joshua T, Hanna, Josiah P, Ferland, Libby, Mattei, Nicholas, & Goldsmith, Judy. 2012. The Academic Advising Planning Domain. *WS-IPC 2012*.
- Guestrin, Carlos, Koller, Daphne, & Parr, Ronald. 2001. Multiagent Planning With Factored MDPs. *Advances In Neural Information Processing Systems (NIPS)*.
- Hoey, Jesse, St-Aubin, Robert, Hu, Alan, & Boutilier, Craig. 1999. SPUDD: Stochastic Planning Using Decision Diagrams. *In: Proceedings Of The Fifteenth Conference On Uncertainty In Artificial Intelligence (UAI)*.
- Hostetler, Jesse, Fern, Alan, & Dietterich, Tom. 2014. State Aggregation In Monte Carlo Tree Search. *In: Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI)*.
- Jong, Nicholas K. 2005. State Abstraction Discovery From Irrelevant State Variables. *In: Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI)*.
- Keller, Thomas, & Helmert, Malte. 2013. Trial-Based Heuristic Tree Search For Finite Horizon MDPs. *In: Twenty-Third International Conference on Automated Planning and Scheduling (ICAPS)*.
- Kocsis, Levente, & Szepesvári, Csaba. 2006. Bandit Based Monte-Carlo Planning. *In: Proceedings of the 17th European Conference on Machine Learning (ECML)*.
- Koller, Daphne, & Parr, Ronald. 2000. Policy Iteration For Factored MDPs. *In: Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence (UAI)*.
- Kolobov, Andrey, Dai, Peng, Mausam, Mausam, & Weld, Daniel S. 2012. Reverse Iterative Deepening for Finite-Horizon MDPs with Large Branching Factors. *In: Twenty-Second International Conference on Automated Planning and Scheduling (ICAPS)*.
- Li, Lihong, Walsh, Thomas J, & Littman, Michael L. 2006. Towards a Unified Theory of State Abstraction for MDPs. *In: Proceedings of the Ninth International Symposium on Artificial Intelligence and Mathematics (ISAIA)*.
- McMahan, H Brendan, Likhachev, Maxim, & Gordon, Geoffrey J. 2005. Bounded Real-Time Dynamic Programming: RTDP with Monotone Upper Bounds and Performance Guarantees. *In: Proceedings of the 22nd International Conference on Machine Learning (ICML)*.
- Puterman, Martin L. 2014. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*.
- Puterman, Martin L, & Shin, Moon Chirl. 1978. Modified Policy Iteration Algorithms for Discounted Markov Decision Problems. *Management Science*.
- Raghavan, Aswin, Joshi, Saket, Fern, Alan, Tadepalli, Prasad, & Khardon, Roni. 2012. Planning in Factored Action Spaces with Symbolic Dynamic Programming. *In: Twenty-Sixth AAAI Conference on Artificial Intelligence (AAAI)*.
- Raghavan, Aswin, Khardon, Roni, Fern, Alan, & Tadepalli, Prasad. 2013. Symbolic Opportunistic Policy Iteration for Factored-Action MDPs. *In: Advances in Neural Information Processing Systems (NIPS)*.
- Sanner, Scott. 2010. Relational Dynamic Influence Diagram Language (RDDL): Language Description. *Unpublished ms. Australian National University*.
- St-Aubin, Robert, Hoey, Jesse, & Boutilier, Craig. 2001. APRI-CODD: Approximate Policy Construction using Decision Diagrams. *Advances in Neural Information Processing Systems (NIPS)*.
- Walsh, Thomas J, Goschin, Sergiu, & Littman, Michael L. 2010. Integrating Sample-Based Planning and Model-Based Reinforcement Learning. *In: Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI)*.

---

# The Survival Filter: Joint Survival Analysis with a Latent Time Series

---

**Rajesh Ranganath**

Computer Science Dept.  
Princeton University  
Princeton, NJ 08540

**Adler Perotte**

Biomedical Informatics Dept.  
Columbia University  
New York, NY 10032

**Noémie Elhadad**

Biomedical Informatics Dept.  
Columbia University  
New York, NY 10032

**David M. Blei**

Computer Science Dept.  
Statistics Dept.  
Columbia University  
New York, NY 10027

## Abstract

Survival analysis is a core task in applied statistics, which models time-to-failure or time-to-event data. In the clinical domain, for example, meaningful events are defined as the onset of different diseases for a given patient. Survival analysis is limited, however, for analyzing modern electronic health records. Patients often have a wide range of diseases, and there are complex interactions among the relative risks of different events. To this end, we develop the survival filter model, a time-series model for joint survival analysis that models multiple patients and multiple diseases. We develop a scalable variational inference algorithm and apply our method to a large data set of longitudinal patient records. The survival filter gives good predictive performance when compared to two baselines and identifies clinically meaningful patterns of disease interaction.

## 1 INTRODUCTION

Electronic health records enable unprecedented opportunity to understand and form predictions about disease [Jensen et al., 2012, Hripcsak and Albers, 2013]. With historical data about the trajectories of millions of patients, we can learn patterns of disease risk and exploit these patterns to provide better care to future patients.

The classical statistics tool for analyzing the progression of a disease is survival analysis, a method that estimates each patient's hazard or risk for a disease in question [Cox, 1972]. Survival analysis is widely used in medical science to characterize and understand the progression of individual diseases [Shepherd et al., 1995, Stupp et al., 2005].

Classical survival analysis, however, cannot accommodate the complex health data that we now have collected; it is only formulated to analyze one disease at a time. In modern electronic health record data, patients often have several

diseases (called “comorbidities”) with complex interactions among them. Specifically, the occurrence of one disease often affects the progression of others. We need new tools to account for this complexity.

Consider the data in Figure 1, where time is in the x-axis. The top panel shows the setting that classical survival analysis requires. All patients begin at the same time, and we measure one disease outcome. (In this case, it is whether the patient is diagnosed with diabetes.) The bottom panel illustrates the real-world setting of electronic health records. Patients begin at different times and we simultaneously measure many different diseases. These data can potentially reveal interactions between patterns of progression, but classical survival analysis cannot provide such inferences. For example, Patient 1 in the bottom panel illustrates that diabetes and hypertension are significant risk factors for developing a myocardial infarction. A traditional survival analysis may be constructed to capture this specific interaction, but cannot simultaneously capture the relationship between risk factors for diabetes (e.g., obesity) and the onset of diabetes.

We build on survival analysis to develop the survival filter, a new probabilistic model for estimating multivariate risk patterns from large-scale electronic health records. The survival filter is a latent-variable time series model of diagnostic codes. Each patient is represented as a sequence of latent variables. At each time point, a patient's hazards for each disease relates to his or her latent representation.

The survival filter can be thought of as a joint survival analysis model where each patient's sequence of latent representations loosely represents his or her state of health. Given a large data set of patients over time—data of the form of the bottom panel of Figure 1—survival filter inference characterizes each patient and captures complex global patterns of interactions for a large set of diseases.

Using the survival filter, we study 13,000 patients from NewYork-Presbyterian Hospital; these data span over 20 years and contain 8,800 types of diagnoses. It scales well to this size of data and uncovers meaningful relationships be-



tween diseases that would otherwise be difficult to identify.

## 2 SURVIVAL ANALYSIS

In this section we review survival analysis and hazards.

**Survival Analysis.** Survival analysis studies the time duration until the occurrence of an event. Example events include failure of a machine, heart attack, and retirement.

Observations in survival analysis have two types. The first type of observation indicates the event has occurred (called “failure”) at a specific time (failure time). The second type indicates the event has not occurred before the observed time. These observations are called censored observations because the true failure time is censored in the observed data. Formally, the observations in survival analysis can be represented as pairs  $(t, c)$  where  $t$  is a time, and  $c$  is a binary value that indicates whether the observation is censored.

The simplest model for survival analysis assumes that the failure times are drawn from some unknown distribution  $F$  over positive values. The setup assumes that all observations are synchronized at their starts. Given this modeling choice, a nonparametric estimate of the CDF of  $F$ , also denoted by  $F$ , is the Kaplan-Meier estimator [Kaplan and Meier, 1958]. The Kaplan-Meier estimator is the nonparametric maximum likelihood estimator of  $1 - F(t)$ , also called the survival function, in the presence of censored data.

The time measurements in survival analysis can be treated as continuous or discrete (e.g., months or years). In this article we will focus discrete survival times.

**Hazards.** An alternative view of survival analysis is through hazard functions. Hazard functions represent the instantaneous chance of failing at time  $t$  given survival up to time  $t$ . In the discrete time setting, the hazard is the conditional probability of failing at time  $t$  given that the failure occurs at time  $t$  or later,

$$h(t) = P(T = t | T \geq t). \quad (1)$$

The Nelson-Aalen [Nelson, 1972] estimator forms the nonparametric maximum likelihood estimator of the sum of hazard function over time (cumulative hazard). The CDF  $F$  of the underlying distribution implied by the hazards is

$$F(t) = 1 - \exp\left(\sum_{s=0}^t h(s)\right).$$

Unlike the cumulative distribution function and survival function, we can specify the hazard function locally in each discrete time block by a number between zero and one. We will use this property when we develop the survival filter.

## 3 THE SURVIVAL FILTER FOR ELECTRONIC HEALTH RECORDS

In this section we first describe electronic health records and corresponding survival problems. We then describe our model, the survival filter.

**Electronic Health Records.** The Electronic Health Record (EHR) comprises all documentation entered for a patient throughout their interactions with a healthcare institution. It contains a wide range of observations through time, ranging from free-text notes authored by clinicians, medication orders, laboratory test results, procedures, demographic information, and diagnosis codes.

Diagnosis codes (also called billing codes) are structured codes from a standard taxonomy, namely the ICD9 hierarchy (International Classification of Diseases, 9th revision). ICD9 codes are used in all healthcare institutions. While the ICD9 hierarchy contains approximately 16,000 codes, in practice about 9,000 of them are commonly documented. After each visit, a clinician assigns each patient a set of ICD9 codes to reflect the diseases or concerns that were taken care of during the visit.

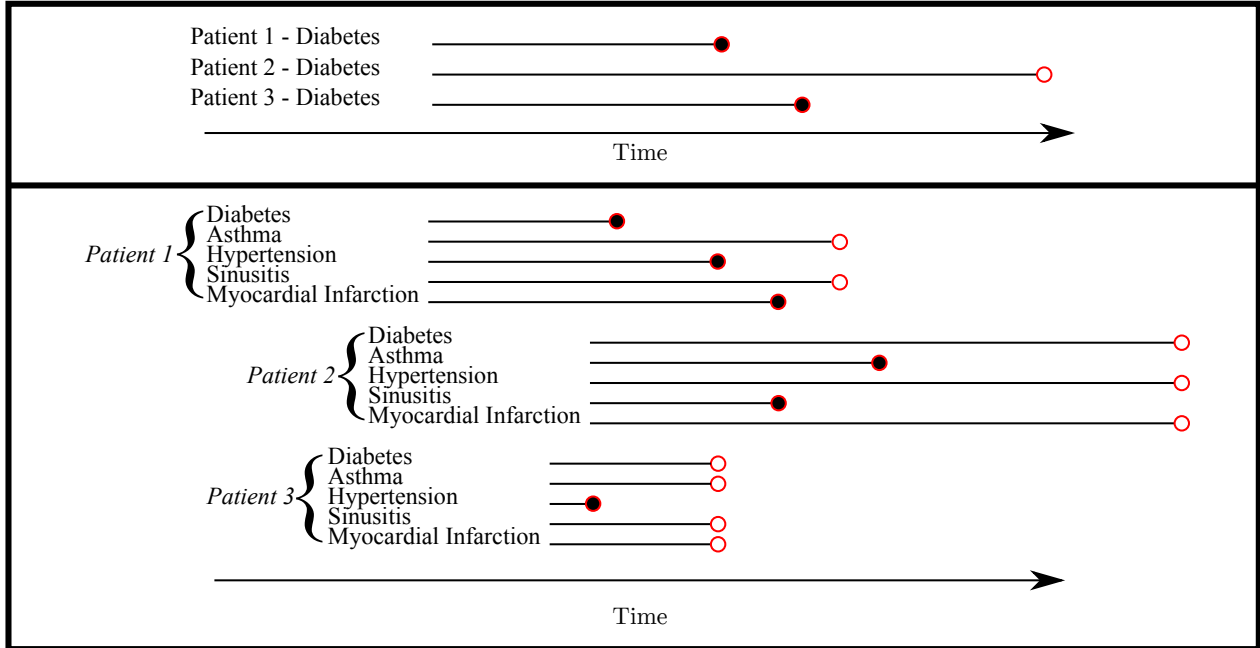
For instance, after a visit to their primary provider, a patient is assigned ICD9 codes for “Essential Hypertension,” and “Diabetes”. At their next visit to their ophthalmologist, the patient can have the ICD9 code for “Nonproliferative Diabetic Retinopathy.” In this example, while the patient has diabetes at both visits, the ICD9 code for diabetes is only observed at the first visit. Furthermore, note that the ICD9 codes are correlated. Retinopathy, an eye condition, is a common complication for patients with diabetes, and hypertension and diabetes are known comorbidities.

EHR data are longitudinal records, represented as a collection of per-patient time series ICD9 codes. Note that an extremely sparse set of ICD9 codes will be used for any given patient.<sup>1</sup> Our goal is to use these data to run a joint survival analysis of every ICD9 code. Specifically, we seek a method that:

- estimates the per-patient risk for all ICD9 codes at any given visit time;
- handles multiple survival problems that are not aligned in time across patients;
- scales to 9,000 ICD9 codes;
- captures interaction between survival problems (e.g., retinopathy, diabetes, and hypertension).

This differs significantly from the setup of traditional survival analysis, where patients are forced to start at the same

<sup>1</sup>In our representation, we use “visit time,” not clock time, as the time unit. Visit time better represents time when studying the temporal course of diseases [Hripcsak et al., 2015].



**Figure 1:** A comparison of standard survival analysis (top frame) and the survival filter (bottom frame). A filled circle represents an observed event, while an empty circle represents a censored one. In the case of standard survival analysis, patients in a cohort are aligned by an event. In the survival filter, patients are not aligned and unlike standard survival analysis, many conditions are considered simultaneously.

time and where we can only analyze a single disease. We now describe the survival filter, a model that addresses these goals to perform large-scale joint survival analysis of EHR.

**Survival Filter.** In the discrete time setting, let the observation pair  $(t, c)$  of time and censoring indicator be represented as a binary vector indexed by the clock with length equal to the observation time. This binary vector has a one in the last entry in the case of failure and is all zero in the case of censoring. We adopt this view of the observations for the survival filter.

Let  $P$  be the number of patients. Let  $n_p$  be the number of time intervals for patient  $p$ , and  $C$  be the number of codes. Then the observation  $x_{p,t,c}$  is one if the code  $c$  is marked in the  $t$ th time interval for patient  $p$  and zero if code  $c$  has not yet occurred for patient  $p$ . To generate this data we propose a latent time series model where each patient has a  $K$  dimensional latent path which along with a  $C \times K$  weight matrix  $W$  produces the hazard for each of the  $C$  codes. Let  $D$  be a distribution over the positive reals. The generative process for this model is

$$\begin{aligned}
 W &\sim D \\
 z_{p,1} &\sim \text{Normal}(z_\mu, \sigma_{z_0}^2) \\
 z_{p,t} &\sim \text{Normal}(z_{p,t-1}, \sigma_z^2) \\
 x_{p,t,c} &\sim \text{Bernoulli}(1 - \exp(-W_c^\top \exp(z_{p,t}))).
 \end{aligned}$$

The hazard of a code  $c$  for patient  $p$  at a time  $t$  is  $1 - \exp(-W_c^\top \exp(z_{p,t}))$ . The positivity of  $W$  and  $\exp(z_{p,t})$  guarantee that  $1 - \exp(-W_c^\top \exp(z_{p,t}))$  is a valid hazard (probability) between zero and one. Larger  $W_{c,k}$  indicate larger hazards for code  $c$  when factor  $k$  is active.

This model handles the criteria described above. First, it provides a simultaneous analysis of all ICD9 codes. Second, it handles misaligned patients by defining the hazards to be a function of the a shared latent space rather than a function of a fixed shared clock  $h(t)$  (as in the classical setting). Third, through the matrix  $W$  it captures the relationship between different survival problems. Specifically, when  $W_{c,k}$  and  $W_{d,k}$  are large, the events  $c$  and  $d$  are more likely to co-occur. Finally, as we will show in Section 4, we can perform efficient computation for the survival filter.

We consider two different priors on  $W$ : the log-normal and the gamma. The gamma distribution is sparsifying when its shape is less than one (this can be seen from the PDF), while the log-normal distribution has a heavier tail. Recent results [Mimno et al., 2014] have shown that log-normal distributions achieve better predictive results and diversity among the weights in a Poisson network model. We derive inference and compare both the log-normal and gamma prior in the experimental section.

**Related Work.** Survival analysis methods have been generalized in many ways beyond the Kaplan-Meier estimator.

One example of such an extension is recurrent event models [Clayton, 1994] which allow for multiple events rather than single events. Cox proportional hazards [Cox, 1972] introduces fixed covariates to scale the patient hazards based on their covariates. Time varying Cox proportional hazard models [Fisher and Lin, 1999] are like Cox proportional hazards, but have a set of covariates that change with time for each patient. Cox-proportional hazard methods are similar to ours in that they define different clocks through the covariates, but differ in that they require covariates and do not capture the relationship between different survival problems. The model with the most similar goal is MEPSUM [Dean et al., 2014]. MEPSUM is a mixture model for multiple kinds of events happening simultaneously. The relationship between events is captured via the latent class label as each latent class contains a nonparametric hazard function for every type of event. Unlike our model, their model assumes that patients are synchronized in time when there are no covariates. Additionally the formulation of their model means that it scales with the number of codes rather than the number of failures, which makes it impractical in large sparse datasets such as those found in electronic health records.

## 4 INFERENCE

The main computational problem in working with the survival filter is computing the posterior distribution of the weights and latent patient trajectory. Computing the posterior of the survival filter analytically is intractable since our likelihood cannot be integrated out. Thus posterior computations require approximations. In this section, we develop a scalable mean field variational inference [Jordan et al., 1999] algorithm to approximate the posterior distribution of the survival filter.

Variational inference transforms the posterior inference problem into an optimization problem. The optimization problem defined by variational inference seeks to find a distribution  $q$  in an approximating family that is close in KL divergence to the posterior distribution. This is equivalent to maximizing the following [Bishop, 2006]:

$$\mathcal{L}(q) := \mathbb{E}_q[\log p(x, z, W) - \log q(z, W)].$$

This function is called the evidence lower bound (ELBO) as it forms a lower bound on  $\log p(x)$ .

**Variational Approximation.** Recall for the survival filter the latent variables are (1)  $z_{pt}$  for all latent states associated with patient  $p$  at time index  $t$  and (2)  $W$ , the matrix shared

across observations. The joint distribution can be written as

$$p(x, z, W) = \prod_{k=1}^K \prod_{c=1}^C p(W_{c,k}) \prod_{p=1}^P p(z_{p,1,k}) \prod_{c:a_{p,1}} p(x_{p,1,c} | z_{p,1}, W) \prod_{t=2}^{n_p} \prod_{k=1}^K p(z_{p,t,k} | z_{p,t-1,k}) \prod_{c:a_{p,t}} p(x_{p,t,c} | z_{p,t}, W).$$

The mean field family posits a variational distribution where the latent variables are independent of each other. Each factorized  $q$  belongs to the same family as in the generative process. Formally, the approximating distribution is

$$q(z, W) = \prod_{k=1}^K \prod_{c=1}^C q(W_{c,k} | \lambda_{c,k}^0, \lambda_{c,k}^1) \prod_{p=1}^P \prod_{t=1}^{n_p} \prod_{k=1}^K p(z_{p,t,k} | \mu_{p,t,k}, \sigma_{p,t,k}^2),$$

where  $\lambda_{c,k_0}$ ,  $\lambda_{c,k_1}$ ,  $\mu_{p,t,k}$ , and  $\sigma_{p,t,k}$  are variational parameters. These variational parameters are then set via an optimization procedure to maximize the ELBO.

**Classical Optimization.** Typical optimization methods for mean field variational inference iteratively optimize the variational parameters associated with each latent variable by holding the others fixed. These updates are easy to derive when the model’s log complete conditional (the log of the distribution of each latent variable conditioned on the rest) has analytic expectation with respect to the variational approximation. This analytic property most commonly occurs in conditionally conjugate exponential families models where the complete conditional is in the exponential family [Ghahramani and Beal, 2001]. Unfortunately, none of the latent variables in our model fall into this class. Instead, we derive a variational algorithm based on sampling from the variational approximation [Salimans and Knowles, 2013, Kingma and Welling, 2014, Rezende et al., 2014, Ranganath et al., 2014, Titsias and Lázaro-Gredilla, 2014].

**Sampling based Variational Inference.** We briefly review stochastic optimization before discussing sampling based variational inference. In the following, we use  $\lambda$  as an example parameter. Let  $\mathcal{L}(\lambda)$  be a function to be maximized and let  $\hat{\nabla}_\lambda \mathcal{L}(\lambda)$  be a draw from a random variable whose expectation is the true gradient  $\nabla_\lambda \mathcal{L}(\lambda)$ . Let  $\rho_t$  be the learning rate, then stochastic optimization updates to the current parameter  $\lambda_t$  can be made with

$$\lambda_{t+1} = \lambda_t + \rho_t \hat{\nabla}_\lambda \mathcal{L}(\lambda).$$

---

**Algorithm 1** Stochastic Variational Inference for the Survival Filter
 

---

**Input:** data  $X$   
**Initialize**  $\lambda$  randomly,  $t = 1$ .  
**repeat**  
 Sample a batch of datapoint  $x_{1...B}$   
**for**  $b = 1...b$  in parallel **do**  
 Use stochastic optimization with Eq. 4 to find the optimal  $\mu_{b,v}$  and  $\sigma_{b,v}$   
**end for**  
 Compute the noisy global gradient for  $\lambda$  (Example: Eq. 8)  
 Update  $\lambda$  with RMSProp  
**until** change in validation metric is small

---

This update converges to a local maximum when the learning rate satisfies the Robbins Monro conditions

$$\sum_{t=1}^{\infty} \rho_t = \infty$$

$$\sum_{t=1}^{\infty} \rho_t^2 < \infty.$$

Stochastic optimization has become a widely used tool in variational inference.

Returning to variational inference, the variational objective is an expectation with respect to the variational approximation. Sampling based variational inference works by writing the gradient of the ELBO as an expectation followed by a stochastic optimization driven by Monte Carlo estimates of the gradient written as an expectation. The gradient as an expectation step comes in two main flavors: (1) those based on transformations and (2) those based on the score function (gradient of the log probability) of the variational approximation. We use both of these techniques to derive an inference algorithm for the survival filter (See the appendix for derivation details).

**Algorithm.** Algorithm 1 summarizes the parallelized stochastic variational inference algorithm we use to approximate posteriors of the variational approximation.

**Scalability** To scale to a large number of censored codes, we need to be able to efficiently compute the likelihood for a patient  $p(x_p)$ . Let  $a_{p,t}$  be the set of codes that have not occurred before time  $t$  for patient  $p$ . Define  $R_{p,t}$  as the relative log likelihood of the failed codes minus the previously failed codes:

$$R_{p,t} = \sum_{c: x_{p,t,c}=1} \log p(x_{p,t,c}=1) - \log p(x_{p,t,c}=0)$$

$$- \sum_{c:[C] \setminus a_{p,t}} \log p(x_{p,t,c}=0).$$

Note that  $R_{p,t}$  can be computed on the order of the number of failures for patient  $p$

By the generative process, we have that the likelihood is

$$\log p(x_p) = \sum_{t=1}^{n_p} \sum_{c \in a_{p,t}}^C \log p(x_{p,t,c})$$

$$= \sum_{t=1}^{n_p} \sum_{c=1}^C \log p(x_{p,t,c}=0) + R_{p,t}$$

$$= \sum_{t=1}^{n_p} \sum_{c=1}^C \log(1 - (1 - \exp(-W_c^\top \exp(z_{p,t})))) + R_{p,t}$$

$$= \sum_{t=1}^{n_p} \sum_{c=1}^C -W_c \exp(z_{p,t}) + R_{p,t}$$

$$= - \left( \sum_{c=1}^C W_c \right) \left( \sum_{t=1}^{n_p} \exp(z_{p,t}) \right) + \sum_{t=1}^{n_p} R_{p,t}. \quad (2)$$

This means that the likelihood for a patient can be computed in time  $O((C + n_p)K + n_p s_p K)$  where  $s_p$  is the number of failures for patient  $p$  instead of of  $O(C n_p K)$ . Thus the runtime scales with the number of uncensored codes rather than by the total number of codes. This efficiency in computing the likelihood will allow for the construction of efficient inference algorithms that scale with the number of failures in the data.

## 5 EMPIRICAL STUDY

In this section, we describe our experimental setup and results.

**Datasets.** Our dataset comprises the longitudinal records of 13,180 patients from a large, metropolitan healthcare institution, NewYorkPresbyterian Hospital. IRB approval was obtained for these experiments. The patient records contain documentation pertaining to all visit types, including outpatient visits, emergency department visits, as well as hospital admissions and intensive care stays (thus with varying ICD9 codes through time for a given patient). The only criteria to include patients in the dataset was at least 5 visits overall to the institutions and among them at least 3 to a primary provider care clinic. We truncated the longitudinal records of patients to 50 visits at most, and thus the mode of the visits was 50. Note that even though the time unit for our analysis is visit, the patient records actually have a wide range of durations (mean 14.5 years; std dev 8 years; median 15.5 years).

For the 13,180 patients, there were overall 8,722 unique ICD9 codes present in at least one visit. On average, each visit had 3.61 ICD9 codes assigned (std dev 2.28; median 3.05), and patients had an average of 189 (std dev 178; median 138) ICD9 codes in their longitudinal records, corre-

sponding to 57 unique codes on average (std dev 36; median 49).

Thus, our dataset represents a large set of patients with a wide range of conditions, as reflected by the large number of ICD9 codes in the dataset.

For our experiments, we held out 100 patient records for validation and parameter tuning and 1,000 patient records for testing purposes.

**Evaluation Metrics.** Standard evaluations in traditional survival analysis rely on concordance; essentially how well can the model rank patients according to the order in which the outcome is observed. This assumes a common clock, or  $t_0$  for all patients, an assumption not held for the survival filter model. Instead, we propose the following three metrics: predictive log likelihood on held out data to assess model fitness, and two metrics well defined in the case of multiple, simultaneous survival analyses. All three metrics are computed by looking forward in the patient time series. We keep the approximate posterior of the shared weights from the training cohort fixed throughout testing.

The first metric computes the log likelihood of all ICD9 codes that have not yet occurred at each visit conditional on all the patient history prior to the visit. Thus, log likelihood is:

$$\log p(x_{c,p,t}) = -W_c^\top z_{p,t-1} \mathcal{I}(x_{c,p,t} = 0) + \log(1 - \exp(-W_c^\top z_{p,t-1})) \mathcal{I}(x_{c,p,t} = 1),$$

where  $\mathcal{I}$  is the indicator function. For each patient in the test set, the predictive log likelihood is computed at each visit after the third visit. Procedurally, this means that we test at visit 4 conditioning on the first three visit, followed by testing at visit 5, conditioning on the first four, and so on.

The second metric computes the Mean Average Ranking of the codes that failed (i.e., first time observed) at visit  $t$  in the set of all ICD9 codes that have not yet failed (i.e., not yet observed) at that visit. The ICD9 ranks are computed by ordering the hazards the model assigns to each code at visit  $t$  based on the patient’s latent state at visit  $t - 1$ .

The third metric is Recall at D (in our experiments, D is set to 10). Recall at D computes how many failed codes (i.e., first time observed) are in the top D codes, as ordered by the hazards assigned by the model to each code.

**Baselines** We consider two baselines for this problem. The first baseline, which we call Mean Disease Risk, considers the frequency of ICD9 codes over the entire population. Given the training set of longitudinal records, a mean hazard is computed for each ICD9 code. Thus, this baseline outputs a fixed hazard prediction through time for any new patient visit.

The second baseline is patient specific, and is called Person Disease Risk. It computes a single hazard for all ICD9 codes

| Factor A  | Factor B  |
|---|---|
| Lumbago<br>Osteoarthritis<br>Myalgia and myositis<br>Pain in joint<br>Pain in limb<br>Backache<br>Arthropathy<br>Pain in joint involving lower leg<br>Cervicalgia<br>Pain in joint involving shoulder region  | Depressive disorder<br>Anxiety state<br>Major depressive disorder, recurrent<br>Major depressive disorder, single episode<br>Dysthymic disorder<br>Adjustment disorder<br>Unknown cause of morbidity or mortality<br>Panic disorder without agoraphobia<br>Unspecified personality disorder<br>Palpitations |
| Factor C  | Factor D  |
| HIV counseling<br>Pregnant state, incidental<br>Vaginitis<br>Special gynecological examination<br>Routine gynecological examination<br>Counseling and advice on contraception<br>Mother with single liveborn<br>Supervision of other normal pregnancy<br>Normal delivery<br>Leiomyoma of uterus | Headache<br>Dizziness and giddiness<br>Migraine<br>Disorder of optic nerve and visual pathways<br>Visual field defect<br>Unspecified endocrine disorder<br>Cushing's syndrome<br>Optic atrophy<br>Neoplasm of endocrine glands<br>Visual discomfort   |

**Figure 3:** Example factors for the survival filter represented by the ICD9 codes with highest hazard for each factor.

based on the empirical frequency of failures at all previous visits. This baseline captures in essence the level of sickness of a patient, as sicker patients experience more code failures (i.e., observe more ICD9 codes).

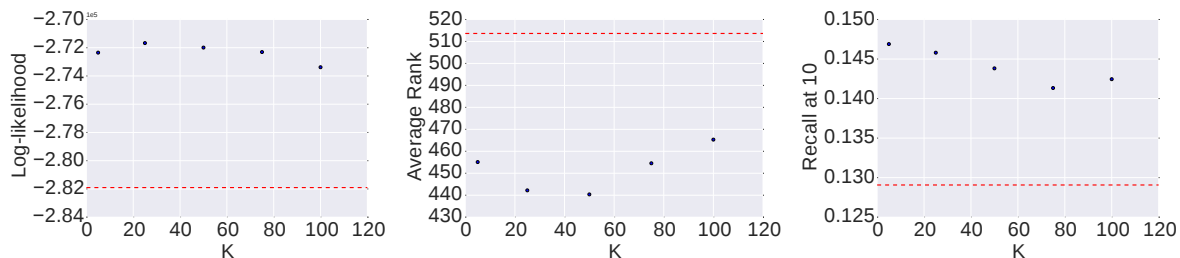
**Hyperparameters.** We set the prior variance on the initial state of the latent trajectories to 10 and the prior mean  $-3$ . The large variance accounts for the fact that patient’s records start at different points. We set the transition variance to  $.1$ . For log-normal weights we set the log scale to  $\log(10^{-10})$ , and the shape to 30. This distribution places a lot of mass near zero. To encourage sparsity of the gamma weights, we set the shape to  $.02$  and the rate to  $0.3$ .

RMSProp also contains a scaling parameter. For the local maximization step we use a decreasing schedule given by  $(1 + t)^{-.8}$  and for the global gradients we set the constant to  $.1$ .

We explore several different sizes for the latent space ranging from  $K = 5$  to  $K = 100$ .

**Results.** Figure 2 plots the evaluation metrics as a function of  $K$  for the log-normal model. We find that the model with  $K = 25$  does best with the best predictive log likelihood and a nearly best performance on Mean Average Ranking. All of the models outperform the plotted mean disease risk baseline on all metrics. We find that the gamma models performs worse than the log-normal model for all  $K$  with a best test log likelihood of  $-284874$ . Finally, all of the models outperform the person disease risk baseline on log likelihood ( $-359079$ ).

Figure 3 displays four of the factors found by the log-normal survival filter with  $K = 25$ . These components represent clinically meaningful groups of conditions.



**Figure 2:** The survival filter (blue dots) outperforms the mean disease risk baseline (dashed red line) for all values of  $K$  on all metrics.

## 6 DISCUSSION

In this paper, we have developed the survival filter, a latent timeseries model for joint survival analysis. The main advantages of the survival include jointly modeling time-to-event data for a large set of events and without specifying alignment across individuals, and an efficient mean field variational inference algorithm that scales in the number of events. We demonstrated the use of the survival filter by measuring the predictive likelihood on a real-world clinical data set and demonstrate superior performance relative to baselines and interpretable latent factors.

The survival filter is a general joint survival analysis model and can be used to study survival problems beyond those in electronic health records. It can be used in any situation where there are multiple simultaneous survival problems that are not necessarily aligned by a true clock. For example, the survival filter can be used to make movie recommendations. In this setting the codes are movies, and the patients are the users. Here, failure of a particular code at time  $t$  means that a movie was watched at time  $t$  and the hazards capture the chance that a movie is watched by a user at time  $t$ .

**Acknowledgements** Rajesh Ranganath is supported by an NDSEG fellowship. David M. Blei is supported by NSF BIGDATA NSF IIS-1247664, ONR N00014-11-1-0651, and DARPA FA8750-14-2-0009. Noémie Elhadad and Adler Perotte are supported by NSF IIS-1344668.

## 7 APPENDIX

**Sample-based gradients** Transformation based approaches [Kingma and Welling, 2014, Rezende et al., 2014, Titsias and Lázaro-Gredilla, 2014] write the ELBO as an expectation with respect to the a standard distribution without variational parameters and moves the differential operator inside of the expectation. Formally let  $r(y)$  be a standard distribution and let  $T$  be a transformation such that  $T(y, \lambda)$  is distributed as  $q_\lambda$ , then the ELBO can be written as

$$\mathcal{L}(\lambda) = \mathbb{E}_r[\log p(x, T(y, \lambda)) - \log q(T(y, \lambda))].$$

When  $T$  and the model and approximation are differentiable the gradient of the ELBO is given by the ELBO as

$$\nabla_\lambda \mathcal{L}(\lambda) = \mathbb{E}_r[\nabla_z [\log p(x, T(y, \lambda)) - \log q(T(y, \lambda))] \nabla_\lambda T].$$

See Kingma and Welling [2014] for a complete derivation of this identity. In our use below,  $r$  will be the standard normal transformation and  $T$  will be  $z = \sigma y + \mu$ .

Score function based approaches [Ranganath et al., 2014, Mnih and Gregor, 2014] are based on the following identity

$$\nabla_\lambda \mathcal{L}(\lambda) = \mathbb{E}_q[\nabla_\lambda \log q(z|\lambda)(\log p(x, z) - \log q(z))]. \quad (3)$$

See Ranganath et al. [2014] for a derivation of this.

The convergence time of stochastic optimization is related to its noise, so in the sequel we derive analytically when possible.

**Gradient for  $q(z_{p,t})$ .** The variational approximation  $q(z_{p,t,k} | \mu_{p,t,k}, \sigma_{p,t,k}^2)$  is a normal distribution with mean  $\mu_{p,t,k}$  and variance  $\sigma_{p,t,k}^2$ . The gradient for the variational parameters can be mostly computed analytically.

$$\begin{aligned} \nabla_{\mu_{p,t,k}} \mathcal{L} &= -\exp(\mu_{p,t,k} + \frac{1}{2}\sigma_{p,t,k}^2) \sum_{c=1}^C \mathbb{E}[W_c] \\ &\quad - \frac{1}{\sigma_z^2} (2\mu_{p,t,k} - \mu_{p,t-1,k} - \mu_{p,t+1,k}) \\ &\quad + \nabla_{\mu_{p,t,k}} \mathbb{E}[R_{p,t}], \\ \nabla_{\sigma_{p,t,k}^2} \mathcal{L} &= -\frac{1}{2} \exp(\mu_{p,t,k} + \frac{1}{2}\sigma_{p,t,k}^2) \sum_{c=1}^C \mathbb{E}[W_c] \\ &\quad - \frac{1}{\sigma_z^2} - \nabla_{\sigma_{p,t,k}^2} \mathbb{E}[R_{p,t}] + \frac{1}{2\sigma_{p,t,k}^2}. \end{aligned} \quad (4)$$

$\mathbb{E}[W_c]$  can be computed analytically for both the gamma and log-normal distribution. Recall the definition of  $R_{p,t}$

$$\begin{aligned} R_{p,t} &= \sum_{c: x_{p,t,c}=1} \log p(x_{p,t,c}=1) - \log p(x_{p,t,c}=0) \\ &\quad - \sum_{c:[C] \setminus a_{p,t}} \log p(x_{p,t,c}=0). \end{aligned}$$

Its expected value is

$$\begin{aligned} \mathbb{E}_q[R_{p,t}] &= \sum_{c:x_{p,t,c}=1} \mathbb{E}[\log p(x_{p,t,c} = 1)] \\ &+ \mathbb{E}[W_c]^\top \mathbb{E}[\exp(z_{p,t})] \\ &+ \sum_{c:[C]\setminus a_{p,t}} \mathbb{E}[W_c]^\top \mathbb{E}[\exp(z_{p,t})]. \end{aligned}$$

Its derivative with respect to its mean is computed by the transformation approach using the identity  $z = y\sqrt{\sigma_{p,t,k}^2} + \mu_{p,t,k}$  where  $y$  is drawn from a standard normal  $\mathcal{N}$ .

$$\begin{aligned} \nabla_{\mu_{p,t,k}} \mathbb{E}_q[R_{p,t}] &= \sum_{c:x_{p,t,c}=1} \mathbb{E}_q(W_c) \mathbb{E}_{y \sim \mathcal{N}} \left[ \frac{W_{c,k} \exp(z)}{\exp(-W_c^\top \exp(z)) - 1} \right] \\ &+ \mathbb{E}[W_{c,k}] \exp(\mu_{p,t,k} + \frac{1}{2}\sigma_{p,t,k}^2) \\ &+ \sum_{c:[C]\setminus a_{p,t}} \mathbb{E}[W_{c,k}] \exp(\mu_{p,t,k} + \frac{1}{2}\sigma_{p,t,k}^2). \end{aligned}$$

Similarly it's derivative with respect to the variance is given as

$$\begin{aligned} \nabla_{\sigma_{p,t,k}^2} \mathbb{E}_q[R_{p,t}] &= \frac{1}{2} \left( \sum_{c:x_{p,t,c}=1} \mathbb{E}_q(W_c) \mathbb{E}_{y \sim \mathcal{N}} \left[ \frac{y W_{c,k} \exp(z)}{\exp(-W_c^\top \exp(z)) - 1} \right] \right. \\ &+ \mathbb{E}[W_{c,k}] \exp(\mu_{p,t,k} + \frac{1}{2}\sigma_{p,t,k}^2) \\ &+ \left. \sum_{c:[C]\setminus a_{p,t}} \mathbb{E}[W_{c,k}] \exp(\mu_{p,t,k} + \frac{1}{2}\sigma_{p,t,k}^2) \right). \end{aligned}$$

We can compute noisy, unbiased estimates of this gradient by sampling from the variational approximation for  $W_{c,k}$  and sampling  $y$  from the standard normal.

The gradient can be computed in time  $O((C + n_p)K + n_p s_p K)$  rather than  $O(Cn_p K)$ . The only portion of the gradient that we cannot compute analytically is the portion associated with failing codes. This portion requires sampling from the variational approximation. This means we can exploit the sparsity of the failures as we only have to sample small fractions of the  $W$  matrix rather than the entire  $W$  matrix. This results in an order of magnitude less samples from the underlying random generator for each noisy gradient and produces lower variance gradients than sampling entirely.<sup>2</sup>

The gradient of the variational parameters of the first and last point can be expressed similarly.

<sup>2</sup>Variance can be a problem in sampling based variational approximations [Kingma and Welling, 2014, Rezende et al., 2014, Ranganath et al., 2014].

**Gradients for Log-Normal  $W$ .** Similar to the time series  $z_{p,t}$ , the only component of the gradient of the variational parameters of  $W$  that is not analytically tractable is due to the failures. To symmetrize this update with the latent time series, we represent the log-normal distribution as an exponentiated normal. That is  $W_{c,k} = \exp(\tilde{W}_{c,k})$ , where  $\tilde{W}_{c,k}$  is normally distributed with mean  $\mu_w$  and variance  $\sigma_w^2$ . In this setup, the variational approximation for  $\tilde{W}_{c,k}$  is normally distributed with variational parameters  $\lambda_{c,k}^0$  (the mean) and  $\lambda_{c,k}^1$  (the variance). The gradient for the mean of the variational approximation is given by

$$\nabla \lambda_{c,k}^0 = -\frac{1}{\sigma_w^2} (\mu_w - W_{c,k}) \quad (5)$$

$$- \exp(\lambda_{c,k}^0 + \frac{1}{2}\lambda_{c,k}^1) \sum_{p=1}^P \sum_{v=1}^{n_p} \mathbb{E}[\exp(z_{p,t,k})] \quad (6)$$

$$+ \nabla \lambda_{c,k}^0 \mathbb{E}[R_{p,t}]. \quad (7)$$

and the gradient of the variance is

$$\begin{aligned} \nabla \lambda_{c,k}^1 &= -\frac{1}{2\sigma_w^2} \\ &- \exp(\lambda_{c,k}^0 + \frac{1}{2}\lambda_{c,k}^1) \sum_{p=1}^P \sum_{v=1}^{n_p} \frac{1}{2} \mathbb{E}[\exp(z_{p,t,k})] \\ &+ \nabla \lambda_{c,k}^1 \mathbb{E}[R_{p,t}] + \frac{1}{2\lambda_{c,k}^1}. \end{aligned}$$

The gradients of  $R$  are symmetric to the time series updates for both the mean and variance parameter.

Note that the gradient can be computed in time proportional to the number of failures ( $O((C + n_p)K + (\sum_{p=1}^P n_p s_p) K)$ ) rather than the number of codes multiplied by the number of visits ( $O(CK \sum_{p=1}^P n_p)$ ) as sum of  $\mathbb{E}[\exp(z_{p,t,k})]$  across all patients and visits can be shared for each code.

**Gradients for Gamma  $W$**  Finally, we consider the gradient of the parameters of the variational approximation of  $W$  when  $W$  is drawn from a gamma distribution in the generative process. In this setup, the variational approximation for each entry in the weight matrix is gamma distributed with shape  $\lambda_{c,k}^0$  and scale  $\lambda_{c,k}^1$ . Similarly the only part of the gradient for this approximation that cannot be computed analytically is due to the failures. The gradient with respect to the shape is

$$\begin{aligned} \nabla \lambda_{c,k}^0 &= -\beta_w \lambda_{c,k}^1 + (\alpha_w - 1) \Psi^{(1)}(\lambda_{c,k}^0) + 1 \\ &+ (1 - \lambda_{c,k}^0) \Psi^{(1)}(\lambda_{c,k}^0) \\ &- \lambda_{c,k}^1 \sum_{p=1}^P \sum_{v=1}^{n_p} \frac{1}{2} \mathbb{E}[\exp(z_{p,t,k})] \\ &+ \nabla \lambda_{c,k}^0 \mathbb{E}[R_{p,t}]. \end{aligned}$$

where  $\Psi$  is the digamma function and  $\Psi^{(1)}$  is its derivative. The gradient with respect to the scale parameter is

$$\begin{aligned} \nabla_{\lambda_{c,k}^1} &= \beta_w \lambda_{c,k}^0 + \frac{(\alpha_w - 1)}{\lambda_{c,k}^1} + \frac{1}{\lambda_{c,k}^0} \\ &\quad - \lambda_{c,k}^0 \sum_{p=1}^P \sum_{v=1}^{n_p} \frac{1}{2} \mathbb{E}[\exp(z_{p,t,k})] \\ &\quad + \nabla_{\lambda_{c,k}^1} \mathbb{E}[R_{p,t}]. \end{aligned}$$

Similar to the log-normal case this gradient scales with number of failures, and the  $\log p(x = 0)$  terms can be computed analytically.

Rather than using transformations to compute gradient of the expected value of  $\log p(x = 1)$ , we use score style gradients. We already know how to evaluate  $\log p(x = 1)$ , so the all we need to compute the gradient is the score function of the gamma distribution for both the shape  $\alpha$  and the scale  $\kappa$ . The score function of the shape is

$$\nabla_{\lambda_{c,k}^0} \log q(w_{c,k}) = -\log(\lambda_{c,k}^1) - \Psi(\lambda_{c,k}^0) + \log(w_{c,k})$$

The score function of the scale is

$$\nabla_{\kappa} \log q(w_{c,k}) = -\frac{\lambda_{c,k}^0}{\lambda_{c,k}^1} + \frac{w_{c,k}}{\lambda_{c,k}^1{}^2}.$$

Plugging this into Eq. 3 and approximating the expectation by Monte Carlo yields a noisy gradient of the ELBO.

**Data subsampling.** Given the noisy gradients just derived, we can use stochastic optimization to maximize the ELBO. This procedure is inefficient in that every single observation has to be iterated over in order to compute the gradient of the variational parameters for shared  $W$ . Another way this procedure is computationally wasteful is that at early iterations work done on all of the local parameters is based on the randomly initialized variational parameters for shared structure. These inefficiencies can be prohibitive when dealing with large datasets or large data instances.

Stochastic variational inference (SVI) [Hoffman et al., 2013] addresses this by using stochastic optimization. SVI works by first identifying local parameters, latent variables associated with a datapoint, and global parameters, shared latent variables. Next a datapoint is sampled, the optimal variational distribution for the local parameters is computed based on the current value of the global parameters, and a noisy gradient based on the sampled data point is computed. SVI generalizes this to drawing batches of datapoints rather than drawing a single datapoint at each update.

In the survival filter the global parameters are the weights and the local parameters are the latent trajectory. For a fixed variational approximation on  $W$ , we compute the optimal local variational parameters by running a stochastic optimization procedure with noisy gradient given by Eq. 4. An

example global gradient for log-normal weights is given by

$$\begin{aligned} &-\frac{1}{\sigma_w} (\mu_w - W_{c,k}) \\ &\quad - \frac{P}{B} (\exp(\lambda_{c,k}^0 + \frac{1}{2} \lambda_{c,k}^1) \sum_{b=1}^B \sum_{v=1}^{n_{p_b}} \mathbb{E}[\exp(z_{p_b,t,k})] \\ &\quad + \nabla_{\lambda_{c,k}^0} \mathbb{E}[R_{p_b,t}]). \end{aligned} \tag{8}$$

where we have reweighted the data term to maintain unbiasedness of the gradient.

Our approach differs from standard SVI in that we use stochastic optimization to compute the optimal local variational parameters. This approach allows differs from the double stochastic approaches in sampling based variational methods [Titsias and Lázaro-Gredilla, 2014, Ranganath et al., 2014] in that we do run a complete maximization procedure for each data point that is sampled rather than simply follow a noisy gradient. This maximization step can be time consuming, so we find the optimal local variational approximation in parallel.

**Learning Rates.** The standard robbins monro learning rate can be challenging to set in that it does not account for varying length scales or different amounts of noise in each gradient of the coordinate. We instead use RMSProp<sup>3</sup> which scales the gradient by the square root of a running average of the squared gradient. This handles varying length scales as multiplying the objective by a constant does not change the step. RMSProp controls for noise as the moving average of the squared gradient is larger when the variance of the gradient is larger.

## References

- C. Bishop. *Pattern Recognition and Machine Learning*. Springer New York., 2006.
- D. Clayton. Some approaches to the analysis of recurrent event data. *Statistical Methods in Medical Research*, 3(3):244–262, 1994.
- D. R. Cox. Regression models and like-tables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 43(2):187–220, 1972.
- D. O. Dean, D. J. Bauer, and M. J. Shanahan. A discrete-time multiple event process survival mixture (MEPSUM) model. *Psychological methods*, 19(2):251, 2014.
- L. D. Fisher and D. Y. Lin. Time-dependent covariates in the Cox proportional-hazards regression model. *Annual review of public health*, 20(1):145–157, 1999.
- Z. Ghahramani and M. Beal. Propagation algorithms for variational Bayesian learning. In *NIPS 13*, pages 507–513, 2001.

<sup>3</sup> [www.cs.toronto.edu/~tijmen/csc321/slides/lecture\\_slides\\_lec6.pdf](http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf)



- M. Hoffman, D. Blei, C. Wang, and J. Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14(1303–1347), 2013.
- G. Hripcsak and D. J. Albers. Next-generation phenotyping of electronic health records. *Journal of the American Medical Informatics Association*, 20(1):117–121, 2013. ISSN 1067-5027. doi: 10.1136/amiajnl-2012-001145.
- G. Hripcsak, D. J. Albers, and A. Perotte. Parameterizing time in electronic health record studies. *Journal of the American Medical Informatics Association*, 2015.
- P. B. Jensen, L. J. Jensen, and B. Søren. Mining electronic health records: towards better research applications and clinical care. *Nature Reviews. Genetics*, 13(6):395–405, 2012. ISSN 1471-0056. doi: 10.1038/nrg3208.
- M. Jordan, Z. Ghahramani, T. Jaakkola, and L. Saul. Introduction to variational methods for graphical models. *Machine Learning*, 37:183–233, 1999.
- E. L. Kaplan and P. Meier. Nonparametric estimation from incomplete observations. *Journal of the American statistical association*, 53(282):457–481, 1958.
- D. Kingma and M. Welling. Auto-encoding variational Bayes. *International Conference on Learning Representations*, 2014.
- D. Mimno, G. Gopalan, and D. Blei. Necessary evil or first choice? Non-conjugate priors and Poisson community models. In *NIPS Workshop on Variational Inference*, 2014.
- A. Mnih and K. Gregor. Neural variational inference and learning in belief networks. In *ICML*, 2014.
- W. Nelson. Theory and applications of hazard plotting for censored failure data. *Technometrics*, 14(4):945–966, 1972.
- R. Ranganath, S. Gerrish, and D. Blei. Black box variational inference. In *International Conference on Artificial Intelligence and Statistics*, 2014.
- D. Rezende, S. Mohamed, and D. Wierstra. Stochastic back-propagation and approximate inference in deep generative models. *ArXiv e-prints*, January 2014.
- T. Salimans and D. Knowles. Fixed-form variational posterior approximation through stochastic linear regression. *Bayesian Analysis*, 8(4):837–882, 2013.
- J. Shepherd, S. M. Cobbe, I. Ford, C. G. Isles, A. R. Lorimer, P. W. Macfarlane, J. H. McKillop, and C. J. Packard. Prevention of coronary heart disease with pravastatin in men with hypercholesterolemia. *New England Journal of Medicine*, 333(20):1301–1308, 1995. doi: 10.1056/NEJM199511163332001. PMID: 7566020.
- R. Stupp, W. P. Mason, M. J. van den Bent, M. Weller, B. Fisher, M. J. B. Taphoorn, K. Belanger, A. A. Brandes, C. Marosi, U. Bogdahn, J. Curschmann, R. C. Janzer, S. K. Ludwin, T. Gorlia, A. Allgeier, D. Lacombe, J. G. Cairncross, E. Eisenhauer, and R. O. Mirimanoff. Radiotherapy plus concomitant and adjuvant temozolomide for glioblastoma. *New England Journal of Medicine*, 352(10):987–996, 2005. doi: 10.1056/NEJMoa043330. PMID: 15758009.
- M. Titsias and M. Lázaro-Gredilla. Doubly stochastic variational Bayes for non-conjugate inference. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1971–1979, 2014.

---

# Communication Efficient Coresets for Empirical Loss Minimization

---

**Sashank J. Reddi**  
Machine Learning Department  
Carnegie Mellon University  
sjakkamr@cs.cmu.edu

**Barnabás Póczos**  
Machine Learning Department  
Carnegie Mellon University  
bapoczos@cs.cmu.edu

**Alex Smola**  
Machine Learning Department  
Carnegie Mellon University  
alex@smola.org

## Abstract

In this paper, we study the problem of empirical loss minimization with  $l_2$ -regularization in distributed settings with significant communication cost. Stochastic gradient descent (SGD) and its variants are popular techniques for solving these problems in large-scale applications. However, the communication cost of these techniques is usually high, thus leading to considerable performance degradation. We introduce a novel approach to reduce the communication cost while retaining good convergence properties. The key to our approach is the construction of a small summary of the data, called *coreset*, at each iteration and solve an easy optimization problem based on the coreset. We present a general framework for analyzing coreset-based optimization and provide interesting insights into existing algorithms from this perspective. We then propose a new coreset construction and provide its convergence analysis for a wide class of problems that include logistic regression and support vector machines. Preliminary experiments show encouraging results for our algorithm on real-world datasets.

## 1 INTRODUCTION

Empirical loss minimization is one of the most fundamental principles in supervised learning. The key idea is to minimize the loss on the training data subject to some regularization on the model that is being learned. More formally, given the training data  $P = \{(x_1, y_1), \dots, (x_n, y_n)\}$  from a probability distribution on  $\mathcal{X} \times \mathcal{Y}$ , we are interested in the following generic optimization problem:

$$\min_w f(w) \equiv \frac{\lambda}{2} \|w\|^2 + \frac{1}{n} \sum_{i=1}^n \ell(x_i, y_i, w) \quad (1)$$

Throughout this paper we assume that  $\ell$  is convex. Furthermore, we assume that  $\mathcal{X} = \mathbb{R}^d$  and  $\mathcal{Y} = \mathbb{R}$ . Note that the objective function above is strongly convex (a function  $f$  is strongly convex with modulus  $\lambda$  if  $f(w) - \frac{\lambda}{2} \|w\|^2$  is a convex function). Problems conforming to Equation (1) include popular supervised learning algorithms like support vector machines and regularized logistic regression. For example, when  $x_i \in \mathbb{R}^d$ ,  $y_i \in \{-1, 1\}$  and  $\ell(x_i, y_i, w) = \log(1 + \exp(-y_i w^\top x_i))$  (logistic loss), the optimization problem in Equation (1) corresponds to regularized logistic regression. The loss function  $\ell$  is not necessarily smooth as in, for example, support vector machines (SVM) where  $\ell(x_i, y_i, w) = \max(0, 1 - y_i w^\top x_i)$  (hinge loss).

Several algorithms have been proposed in the literature for solving optimization problems of the aforementioned form. We will briefly review a few key approaches in Section 2; however, the algorithms are either largely synchronous or communication intensive. For example, one of the popular approaches for solving such optimization problems is stochastic subgradient descent. At each iteration of the algorithm, a single training example is chosen at random and used to determine the subgradient of the objective function. While such an approach reduces the computation complexity at each iteration, the communication cost is prohibitively expensive in distributed environment.

In this paper, we study the problem described in Equation (1) in the setting where the data is distributed across nodes and hence, communication is expensive in comparison to the computation time. *The main theme of this paper is to reduce communication cost by constructing and optimizing over a small summary of the training data — which acts as a proxy for the entire data set.* Such a summary of the training points is called a *coreset*. While this methodology has been successfully applied to data clustering problems like k-means and k-median (we refer the reader to [4, 5] for a comprehensive survey), it remains largely unexplored for supervised learning and optimization problems. The goal of this paper is to advance the frontier in this direction. In light of the above, the primary contributions of this paper are as follows:

- We describe a general framework for designing coreset-based algorithms. This also provides insights into existing algorithms from the coresets viewpoint.
- We propose a novel coreset-based algorithm with low communication cost and provable guarantees on the convergence to the optimal solution.
- We demonstrate the efficiency of the proposed algorithm on a few real-world datasets. In particular, we show that the proposed approach reduces the communication cost significantly.

Our paper is structured as follows. We begin with a discussion on the related work in Section 2. In Section 3, we describe a general framework for the coreset-based methodology. We then propose a coreset-based algorithm in Section 4 and provide its convergence analysis. We finally conclude by demonstrating the empirical performance of the algorithm in Section 5.

## 2 RELATED WORK

As noted earlier, problem in Equation (1) arises frequently in the machine learning and optimization literatures and hence has been a subject of extensive research. Consequently, we cannot hope to do full justice to all the related work. We instead mention the key relevant works here and refer the reader to the appropriate references for a more thorough coverage.

**First-order methods:** In large-scale machine learning and convex optimization applications, first-order methods are popular due to their cheap iteration cost. The classic approach in first order methods is the gradient descent approach. For strongly convex functions  $f$  with  $L$ -lipschitz gradient, gradient descent has linear convergence rate i.e.,  $f(w_t) - f(w_*) \leq \epsilon$  in  $O(\log(1/\epsilon))$  iterations where  $w_t$  and  $w_*$  are the  $t^{\text{th}}$  iterate of gradient descent and the optimal solution respectively [10]. The constants can be further improved by the means of accelerating techniques [10]. On the other hand, when  $\ell$  is non-smooth, gradient descent methods have sub-linear convergence rates.

While gradient descent methods have appealing convergence properties, they have two major shortcomings: (1) they require evaluation of  $n$  gradients at each iteration, typically leading to high computational cost, and (2) the communication costs is also high. A popular modification of this algorithm in large-scale settings is the stochastic gradient descent. While the computational cost per iteration decreases, the linear convergence property is lost. This is due to the variance introduced by stochasticity of the approach. Recently, there has been a surge in interest to address this issue by incremental methods (see [13, 7]). By reducing the variance, these approaches achieve low iteration complexity while retaining the good convergence properties. How-

ever, all these approaches still do not address the other major shortcoming — namely, high communication cost.

**Active Set & Cutting plane Methods:** Our approach is also related to the classic active set and cutting plane methods used in the optimization and the machine learning literatures [11]. The basic idea is to find a *working set* of constraints, i.e., those inequality constraints of the optimization problem that are either fulfilled with equality or are otherwise important to the optimization problem. These methods are particularly popular in the SVM literature. Scheinberg et al. [12] and Joachims et al. [6] provide more details on these approaches. However, these approaches are inherently sequential and not communication friendly. Moreover, it is observed that these approaches are typically outperformed by subgradient methods [14]. While the basic theme of these methods is similar to that of ours insofar that we compute a similar *summary* of the training data at each iteration, the key distinction is the approach and methodology used in constructing the summary. Moreover, our approach is much more general and can be applied to a wide range of loss functions.

**Coresets:** Our approach is closely related to the paradigm of coresets used in the theory literature [2, 4, 5]. The basic idea of coresets is to extract a small amount of relevant information from the given data and work on this extracted data. Coresets have been proposed on a variety of data clustering problems such as  $k$ -means,  $k$ -medians, and projective clustering. This approach is particularly important for NP-hard problems like  $k$ -means. For example, coresets of size  $O(k/\epsilon^4)$  and independent of  $n$  (number of the data points) have been proposed for the  $k$ -means problem [2, 4]. If  $k$  is small, such an approach makes it possible to find optimal solution of  $k$ -means simply by an exhaustive search. Furthermore, coresets can seamlessly handle distributed and streaming settings and hence, are suitable to large-scale real-world applications. We refer the reader to the excellent (but outdated) survey on coresets [1] for more details. Recently, a unifying coreset framework has been proposed for data clustering problems [4], which provides a more comprehensive treatment; interested readers may also refer to the references therein. While there has been some progress in borrowing ideas from coresets in the context of SVMs [15], this intersection remains largely unexplored.

**Distributed Methods:** Owing to large-scale machine learning applications, there has been a recent surge of interest in distributed training of models. The basic idea is to solve subproblems in parallel, followed by averaging at each iteration. For example, [17, 9] propose an algorithm with a trade-off between computation and communication costs. The Alternating Direction Method of Multipliers (ADMM) [3] and its variants are also popular approaches that fall in this category. However, these strategies are either synchronous and communication unfriendly since no communication occurs during the compu-

tation phase. Mini-batch approaches have received considerable attention recently. We refer the reader to [8] and references therein for a more thorough analysis of mini-batch approaches based on stochastic gradient descent.

### 3 A GENERAL FRAMEWORK

We describe our general methodology in this section. Before delving into the details of the framework, we introduce a few definitions and notations in order to simplify our exposition. We denote the objective function in Equation (1) by  $f(w; P)$ . Recall that  $P$  is the training set. The optimal solution of Equation (1) is denoted by  $w_*$  i.e.,

$$w_* = \operatorname{argmin}_w f(w; P) \equiv \frac{\lambda}{2} \|w\|^2 + \frac{1}{n} \sum_{i=1}^n \ell(x_i, y_i, w).$$

We use  $R_*$  to denote  $\|w_* - w_0\|$ , the distance of optimal solution from the initial point. Next, we define the key ingredient in our approach.

**Definition 1.** (Coreset) For given functions  $f$  and  $g$ , we call a set  $C$  an  $\epsilon$ -coreset of  $P$  on a set  $\Omega$  if  $|g(w; C) - f(w; P)| \leq \epsilon$  for all  $w \in \Omega$ .

Note that the above definition is slightly different from the one typically used in the coreset literature (see [4]) in two ways: (i) the set  $C$  is not necessarily a subset of  $P$ . In particular, when the function is of the form described in Equation 1, typically, coresets are constructed for the specific function of  $g$  being the weighted sum of loss and coreset  $C$  being a subset of  $P$ . However, this is not necessarily the case here. (ii) the coreset is restricted to the domain  $w \in \Omega$ . Another noteworthy point is that while coresets are classically defined as a multiplicative approximation, we use the notion of additive approximation. All these relaxations allow us to view other related algorithms through the lens of coresets. The key desirable property of a coreset is that the cardinality of the set  $C$  is small, which will help us reduce the overall communication complexity of the algorithm. For brevity, we drop  $C$  and  $P$  from the notations  $g(w; C)$  and  $f(w; P)$  respectively whenever  $C$  and  $P$  are clear from the context.

With this background we are ready to state our algorithm. At each iteration of the algorithm, the key component of our framework is to compute a new coreset-based on the current solution and solve the optimization problem based on that coreset. The pseudocode is given as Algorithm 1.

First, we note that algorithm is still abstract because it does not specify details about the function  $g_{t-1}(w; C_{t-1}, w_{t-1})$  and coreset  $C_{t-1}$ . Furthermore, feasible region of the subproblem  $\Omega(w_{t-1}, R_{t-1})$  at each iteration is unspecified. These details depend on the specific coreset construction and hence, are explained during the description of the coreset. We now state a general result on performance of Algorithm 1 based on some important properties of the coreset.

---

#### Algorithm 1 Generic Iterative Coreset Algorithm

---

**INPUT:** Initial  $w_0$ , coefficients  $\{\gamma_1, \dots, \gamma_T\}$

- 1: **for**  $t = 1$  to  $T$  **do**
- 2:   Compute the coreset  $C_{t-1}$  with the corresponding function  $g_{t-1}(w; C_{t-1}, w_{t-1})$
- 3:   Solve the following subproblem

$$w_t = \operatorname{argmin}_{w \in \Omega(w_{t-1}, R_{t-1})} g_{t-1}(w; C_{t-1}, w_{t-1})$$

- 4:    $R_t = \gamma_t \cdot R_{t-1}$
  - 5: **end for**
- 

For ease of analysis, throughout the paper, we assume that  $\gamma_t$  is chosen in such a way that  $R_t = \|w_t - w_*\|$ . The analysis for the case where  $R_t$  is an upper bound on  $\|w_t - w_*\|$  is similar.

**Theorem 1.** Suppose we have the following conditions on the function  $g_{t-1}$  for  $1 \leq t \leq T$ :

1.  $g_{t-1}$  is an upper bound on  $f$  and is strongly convex with modulus  $\lambda'_{t-1}$ , for some  $\lambda'_{t-1} > 0$ .
2. The feasible region  $\Omega(w_{t-1}, R_{t-1})$  is convex and contains the optimal solution  $w_*$ .
3.  $C_{t-1}$  is an  $\Delta_{t-1}$ -coreset of  $P$  on  $\Omega(w_{t-1}, R_{t-1})$  with respect to functions  $g_{t-1}$  and  $f$ . More precisely, we need  $g_{t-1}(w; C_{t-1}, w_{t-1}) \leq f(w; P) + \Delta_{t-1}$  for all  $w \in \Omega(w_{t-1}, R_{t-1})$ .

Then for the iterates  $\{w_t\}_{t=1}^T$  of Algorithm 1 we have,

$$R_t = \|w_t - w_*\| \leq \sqrt{\frac{2\Delta_{t-1}}{\lambda + \lambda'_{t-1}}}.$$

*Proof.* We have the following inequalities:

$$\begin{aligned} g_{t-1}(w_*) &\leq f(w_*) + \Delta_{t-1}, \\ g_{t-1}(w_t) + \langle \partial g_{t-1}(w_t), w_* - w_t \rangle \\ &\quad + \frac{\lambda'_{t-1}}{2} \|w_t - w_*\|^2 \leq g_{t-1}(w_*) \end{aligned}$$

The first inequality follows from condition 3 of the theorem. The second inequality follows from the fact that  $g_{t-1}$  is strongly convex with modulus  $\lambda'_{t-1}$  (condition 1). Adding the above two inequalities we get

$$\begin{aligned} g_{t-1}(w_t) + \langle \partial g_{t-1}(w_t), w_* - w_t \rangle \\ + \frac{\lambda'_{t-1}}{2} \|w_t - w_*\|^2 \leq f(w_*) + \Delta_{t-1}. \end{aligned} \tag{2}$$

Because  $f$  is strongly convex with modulus  $\lambda$ , we have

$$f(w_*) + \langle \partial f(w_*), w_t - w_* \rangle + \frac{\lambda}{2} \|w_t - w_*\|^2 \leq f(w_t).$$

Combining it with the fact that  $g_{t-1}$  is an upper bound on function  $f$  (condition 1), we have

$$f(w_*) + \langle \partial f(w_*), w_t - w_* \rangle + \frac{\lambda}{2} \|w_t - w_*\|^2 \leq g_{t-1}(w_t). \quad (3)$$

Adding Equations (2) and (3), we get the following.

$$\langle \partial g_{t-1}(w_t), w_* - w_t \rangle + \langle \partial f(w_*), w_t - w_* \rangle + \frac{(\lambda + \lambda'_{t-1})}{2} \|w_t - w_*\|^2 \leq \Delta_{t-1} \quad (4)$$

To complete the proof we need the following intermediate result.

**Lemma 1.** *Suppose  $g_{t-1}$  satisfies the conditions in Theorem 1, then for iterates  $w_t$ , for  $1 \leq t \leq T$ , of Algorithm 1 we have*

$$\langle \partial g_{t-1}(w_t), w_* - w_t \rangle \geq 0 \quad (5)$$

$$\langle \partial f(w_*), w_t - w_* \rangle \geq 0 \quad (6)$$

*Proof.* We prove the inequality in Equation (5). The inequality in Equation (6) can be proved in a similar manner. Let  $A = \Omega(w_{t-1}, R_{t-1})$  and  $\mathbb{I}_A : \mathbb{R}^d \rightarrow \mathbb{R}^+$  be the indicator function corresponding to  $A$  i.e.,

$$\mathbb{I}_A(w) = \begin{cases} 0 & \text{if } w \in A \\ +\infty & \text{if } w \notin A \end{cases}$$

Recall that the  $w_t$  is the optimal solution of the following:

$$w_t = \operatorname{argmin}_{w \in A} g_{t-1}(w).$$

From the optimality condition of  $w_t$ , we have  $\partial g_{t-1}(w_t) + \partial \mathbb{I}_A(w_t) = 0$ . Therefore, we have

$$\langle \partial g_{t-1}(w_t), w_* - w_t \rangle = \langle -\partial \mathbb{I}_A(w_t), w_* - w_t \rangle \quad (7)$$

Since  $A$  is convex (condition 2 of Theorem 1), the subgradient will be the normal cone of  $A$ . Using the fact that  $w_*, w_t \in A$  (condition 2 of Theorem (1)) and from the definition of the normal cone, we have

$$\langle -\partial \mathbb{I}_A(w_t), w_* - w_t \rangle \geq 0.$$

Using the above inequality in Equation (7), we get the required result.  $\square$

Using the inequalities from Lemma 1 in Equation (4) it is easy to see that the result follows.  $\square$

The above result gives an upper bound on the distance of the iterate  $w_t$  in Algorithm 1 from the optimal solution  $w_*$ . Note that the bound depends on  $\Delta_{t-1}$  which in turn typically depends on the optimality of  $w_{t-1}$ . It is easy to see that convergence to the optimal solution is possible as long

as  $\lim_{t \rightarrow \infty} \Delta_t = 0$ . It is also worth noting that result does not assume anything on the size of the coreset  $C_t$ . However, as we shall see, the communication and computation complexity of the algorithm will critically depend on  $|C_t|$  at each iteration.

Before discussing our algorithm based on this framework, we consider a popular instantiation of this framework — gradient descent. For this discussion, we assume that the loss function  $\ell$  is differentiable and has  $L$ -lipschitz gradient i.e.,  $\|\partial \ell(x_i, y_i, w) - \partial \ell(x_i, y_i, w')\| \leq L\|w - w'\|$ . This smoothness condition on the gradient gives us the following useful result.

**Lemma 2.** [10] *For any function  $h : \mathbb{R}^d \rightarrow \mathbb{R}$  with  $L$ -Lipschitz continuous gradient  $\partial h$ , we have*

$$h(x) \leq h(y) + \langle \partial h(y), x - y \rangle + \frac{L}{2} \|x - y\|^2, \quad \forall x, y \in \mathbb{R}^d.$$

The update for gradient descent is the following:

$$w_{t+1} = w_t - \gamma \partial f(w; P), \quad (8)$$

where  $\gamma$  is the learning rate and is typically set to  $1/L$ . Such an update can be obtained by minimizing the upper bound on  $f$  in Lemma 2. We briefly explain how gradient descent like method fits our framework. We choose the coreset<sup>1</sup>  $C_t = \partial f(w_t; P)$  and the function  $g_t$  as follows:

$$g_t(w; C_t, w_t) = f(w_t; P) + \langle \partial f(w_t), w - w_t \rangle + \frac{L + \lambda}{2} \|w - w_t\|^2 \quad (9)$$

First note that the function  $g_t$  is an upper bound of  $f$  and is strongly convex with modulus  $L + \lambda$ . This can be obtained from Lemma 2. Hence,  $g_t$  satisfies condition 1 of Theorem 1. Next, we set  $\Omega(w_t, R_t) = \mathcal{B}(w_t, R_t)$  where  $\mathcal{B}(w, R)$  represents a ball of radius  $R$  centered around  $w$ . Since this is convex and  $R_t = \|w_t - w_*\|$ , it is easy to see that condition 2 of Theorem 1 holds.

Finally,  $g_{t-1}$  is an  $\Delta_{t-1}$ -coreset with  $\Delta_{t-1} \leq LR_{t-1}^2/2$ . This can be obtained by a straightforward reasoning based on the Taylor expansion of  $f$  and Lemma 2. Thus all the conditions of Theorem 1 hold. Hence, using Theorem 1 we obtain the following corollary.

**Corollary 1.** *The iterates  $w_t$  of gradient descent algorithm like algorithm (minimizing upper bound in Equation 9 subject to the constraint on  $w \in \Omega(w_t, R_t)$ ) satisfy*

$$R_t = \|w_t - w_*\| \leq \sqrt{\frac{2LR_{t-1}^2}{2(L + 2\lambda)}} = R_{t-1} \sqrt{\frac{1}{(1 + \frac{2\lambda}{L})}}.$$

<sup>1</sup>Recall that the coreset could be any summary of the data, and not necessarily one of its subsets.

In general, dropping the constraint that  $w \in \Omega$ , recovers the gradient descent algorithm. The above corollary reproduces the well-known linear convergence rate for gradient descent [10]. Note the dependence of the convergence rate on the condition number  $L/\lambda$ . While the result does not lead to any new convergence rates, it provides an interesting insight that gradient descent can be viewed as solving an optimization problem on a *coreset* based on the gradients at each iteration. However, it is important to note that the communication cost is still high when the condition number is large since the gradient needs to be communicated at each iteration. Hence, gradient descent is not suitable for settings of our interest — that is, distributed settings where communication is expensive.

A natural question that arises is whether we can construct more interesting coresets than the gradients of the function. We provide an affirmative answer to this question in the next few sections.

## 4 CORESET ALGORITHM

In this section, we propose a new coreset-based algorithm. Before discussing the details of the coreset contribution, it is worth mentioning two additional assumptions; however, we should emphasize that the first assumption is only for the ease of exposition.

1. The loss function  $\ell$  is of the form  $\ell(x_i, y_i, w) = \ell(y_i w^\top x_i)$ . Note the slight abuse of notation in the usage of  $\ell$ . In what follows, the quantity  $y_i w^\top x_i$  is referred to as *margin*.
2.  $\ell$  is  $L$ -lipschitz continuous i.e.,  $|\ell(y_i w^\top x_i) - \ell(y_i w'^\top x_i)| \leq L|y_i w^\top x_i - y_i w'^\top x_i|$ .

Loss functions that satisfy the above properties include popular choices such as logistic loss (used in logistic regression) and hinge loss (used in SVM). The significance of these assumptions will become clear as we proceed. We also need the following definitions for our discussion.

**Definition 2.** (Cover) We call a set of points  $S$  as  $\epsilon$ -cover of a set of points  $Q$  if for all  $q \in Q$  there exists a point  $s \in S$  such that  $\|s - q\| \leq \epsilon$ .

Let  $N(x)$  for a point  $x$  in the cover denote the set of points in  $Q$  that are closer to  $x$  than any other point in the cover  $S$ . With slight abuse of notation, let  $\epsilon(Q) = [S, \beta]$ , where  $S$  is an  $\epsilon$ -cover of  $Q$ , and  $\beta$  is the vector of cardinalities of the sets  $\{N(x)|x \in S\}$ . Note that  $\|\beta\|_1 = |Q|$ .

The key insight to our coreset construction of the algorithm is that typically at each iteration there exist only a few *important* data points that are critical from the optimization perspective. For example, consider an iterative algorithm for SVMs. Intuitively, at each iteration, the points that are

close to the margin are crucial in comparison to those away from it. Furthermore, due to the piecewise linear nature of the hinge loss, the points far away from the margin can be represented by a linear function precisely. Hence, it is possible to obtain a good summary of the data through few points near the margin and a linear function. With this intuition, we now present our coreset construction.

We define the set  $P' = \{x'_i\}_{i=1}^n$  where  $x'_i = y_i x_i$ . Our coreset construction consists of two primary steps:

**Step 1:** Identify points whose loss can be approximated by a linear function and construct a *single* linear function as a coreset for these points. Generally, these are points where gradient approximation is good. We denote such a function by `LINEARAPPROX`. The description of this function will depend on  $\ell$ .

**Step 2:** Construct a cover or equivalent functional approximation for the rest of the points in the set  $P'$ . Since  $\ell$  is assumed to be lipschitz, such a cover also provides approximation guarantees on the empirical loss on  $P'$ .

It should be emphasized that while we use the concept of cover for simplicity, a similar analysis can be carried out for clustering based algorithms. In fact, as we will see later, all our experiments are based on clustering. At each iteration, we use disjoint sets  $G_t$  and  $E_t$  to denote the points concerned with these two steps respectively. Note that  $G_t \cup E_t = P'$ . We use  $l_t \in \mathbb{R}^d$  to denote the linear approximation of loss for points in  $G_t$ . Our coreset is  $C_t = (I_t, \beta_t, l_t)$  where  $[I_t, \beta_t] = \epsilon_t(E_t)$  and function  $g_t$  in Algorithm 1 is as follows.<sup>2</sup>

$$g_t(w; C_t) = \frac{\lambda}{2} \|w\|^2 + \frac{1}{n} \left( \sum_{x_e \in I_t} \beta_t^e \ell(w^\top x_e) + w^\top l_t \right) + h_t \tag{10}$$

where  $h_t = (2LR_*|E_t|\epsilon + |G_t|\delta + c)/n$  for some  $\delta > 0$  and constant  $c$ . The pseudocode, based on the above key steps, is given as Algorithm 2. The size of the coreset  $C_t$  depends on the cardinality of set  $E_t$ .

For simplicity, we assume that  $w_0 = 0$  for our analysis. In this case, we have  $R_* = \|w_*\|$ . One of the key components of Algorithm 2 is the function `LINEARAPPROX`. As mentioned earlier, in general, this function depends on the loss function  $\ell$ . We choose  $\Omega(w_0, R_0) = \mathcal{B}(w_0, R_0)$  and  $\Omega(w_t, R_t) = \Omega(w_{t-1}, R_{t-1}) \cap \mathcal{B}(w_t, R_t)$  for  $t \in [1, \dots, T-1]$ . Recall that we assume the coefficients  $\{\gamma_1, \dots, \gamma_T\}$  are chosen in a way such that  $R_t = \|w_t - w_*\|$ . We prove the following result for Algorithm 2. The proof of Theorem 2 relies on result of the generic coreset algorithm, Theorem 1.

**Theorem 2.** Suppose  $g_t$  is as defined in Equation (10) and

<sup>2</sup>Hereinafter we include the parameter  $w_t$  in the coreset description  $C_t$ .

---

**Algorithm 2** Iterative Coreset Algorithm
 

---

**INPUT:** Initial  $w_0$ , coefficients  $\{\gamma_1, \dots, \gamma_T\}$  and  $\{\epsilon_1, \dots, \epsilon_T\}$

- 1: **for**  $t = 1$  to  $T$  **do**
- 2:    $[G_{t-1}, l_{t-1}] = \text{LINEARAPPROX}(P', w_{t-1}, R_{t-1})$
- 3:    $[I_{t-1}, \beta_{t-1}] = \epsilon_{t-1}(P' \setminus G_{t-1})$
- 4:   Coreset  $C_{t-1} = (I_{t-1}, \beta_{t-1}, l_{t-1})$
- 5:   Solve the following subproblem

$$w_t = \underset{w \in \Omega(w_{t-1}, R_{t-1})}{\text{argmin}} g_{t-1}(w; C_{t-1})$$

- 6:    $R_t = \gamma_t \cdot R_{t-1}$
  - 7: **end for**
- 

LINEARAPPROX satisfies the following condition:

$$\max_{w \in \mathcal{B}(w_t, R_t)} \left| \sum_{x_g \in G_t} \ell(w^\top x_g) - [w^\top l_t + c] \right| \leq |G_t| \delta \quad (11)$$

where  $[G_t, l_t] = \text{LINEARAPPROX}(P', w_t)$  and  $c \in \mathbb{R}^d$  and for all  $t \in \{0, \dots, T-1\}$ , then we have

$$R_{t+1} = \|w_{t+1} - w_*\| \leq \sqrt{\frac{2LR_*|E_t|\epsilon_t + |G_t|\delta}{\lambda n}}.$$

*Proof.* We first observe that  $g_t$  (in Equation (10)) is strongly convex with modulus  $\lambda$ . Moreover,  $g_t$  is an upper bound on  $f$  due to the following relation:

$$\begin{aligned} & \frac{1}{n} \left( \sum_{x_e \in I_t} \beta_t^e \ell(w^\top x_e) + w^\top l_t \right) + h_t \\ &= \frac{1}{n} \left( \sum_{x_e \in I_t} \beta_t^e \ell(w^\top x_e) + w^\top l_t + 2LR_*|E_t|\epsilon_t + |G_t|\delta + c \right) \\ &\geq \frac{1}{n} \left( \sum_{x_e \in I_t} \beta_t^e \ell(w^\top x_e) + 2LR_*|E_t|\epsilon_t + \sum_{x_g \in G_t} \ell(w^\top x_g) \right) \\ &\geq \frac{1}{n} \left( \sum_{x_p \in E_t} \ell(w^\top x_p) + \sum_{x_g \in G_t} \ell(w^\top x_g) \right) = \frac{1}{n} \sum_{x \in P'} \ell(w^\top x) \end{aligned}$$

The first inequality follows from the definition of  $h_t$ . The second step follows from the condition on LINEARAPPROX in the theorem statement. The third step follows from the fact that  $\ell$  is  $L$ -lipschitz continuous and  $\|\beta_t\|_1 = |E_t|$ . Combining the above with regularization term proves the fact that  $g_t$  is an upper bound on  $f$ .

It is easy to see that the feasible region  $\mathcal{B}(w_t, R_t)$  is convex and contains the optimal solution  $w_*$ . To obtain an upper

bound on the function  $g_t$ , we observe the following:

$$\begin{aligned} & \frac{1}{n} \left( \sum_{x_e \in I_t} \beta_t^e \ell(w^\top x_e) + w^\top l_t \right) + h_t \\ &\leq \frac{1}{n} \left( \sum_{x_e \in I_t} \beta_t^e \ell(w^\top x_e) + 2LR_*|E_t|\epsilon_t \right. \\ &\quad \left. + \sum_{x_g \in G_t} \ell(w^\top x_g) + 2|G_t|\delta \right) \\ &= \frac{1}{n} \left( \sum_{x \in P'} \ell(w^\top x) + 4LR_*|E_t|\epsilon_t + 2|G_t|\delta \right) \end{aligned}$$

The first and second inequalities follow from the condition of the theorem statement and the lipschitz continuous nature of the loss function  $\ell$ .

Therefore,  $C_t$  with the corresponding function  $g_t$  is an  $\Delta_t$ -coreset where  $\Delta_t \leq (4LR_*|E_t|\epsilon_t + 2|G_t|\delta)/n$ . The above reasoning shows that the function  $g_t$  satisfies all the conditions of Theorem 1. Applying Theorem 1 on the function  $g_t$ , we get the required result.  $\square$

It can be observed that the conditions  $\epsilon_T \rightarrow 0$  and  $\delta \rightarrow 0$  as  $T \rightarrow \infty$  ensure convergence of the algorithm to the optimal solution. In general, we can guarantee that our solution is arbitrary close to the optimal solution by choosing  $\delta$  and  $\epsilon_t$  appropriately. Furthermore, we can ensure linear convergence of our algorithm by decreasing  $\epsilon_t$  by a constant factor at each iteration.

It is also important to study the coreset size and the design choice of  $\epsilon_t$  and  $\delta$  since they determine the communication cost of our algorithm. Let  $\delta = 2LR_* \min\{\epsilon_1, \dots, \epsilon_T\}$ . For this value of  $\delta$ , we observe the following:

1. The size of the coreset depends on the cardinality of  $G_t$ . In general, larger the cardinality of  $G_t$ , smaller is the size of the coreset. Furthermore, as a general rule of thumb, if  $\ell$  is asymptotically linear i.e.,  $\lim_{|m| \rightarrow \infty} |\ell(m) - (cm + d)| = 0$  for some constants  $c, d$ , the performance of our algorithm will depend on the rate of asymptotic linearity.
2. We typically require  $\epsilon_{t+1} \leq \epsilon_t$  for all  $t \in \{0, \dots, T-1\}$ . With such a choice, if  $|G_t|$  does not decrease, size of the coreset may increase. However, observe that  $R_t$  decreases. Thus, typically more points satisfy Equation (11), and the cardinality of  $G_t$  usually decreases.
3. Suppose a subset of  $P'$  satisfies Equation (11) at iteration  $t$  then it will always satisfy the condition in future iterations. This is due to the fact that feasible region shrinks at each iteration. Hence, size of the coreset is always non-increasing.

While the above remarks provide informal reasoning for the size of the coreset, it does not provide a formal analysis.

In order to gain a better understanding, we discuss the implementation of this algorithm and provide a more formal analysis in the case of logistic regression and SVMs. To this end, let us first discuss the function LINEARAPPROX for specific cases.

**LINEARAPPROX for differentiable loss functions:** The linear approximation in the differentiable case can be obtained through the first-order Taylor expansion of the loss function. More formally, we have

$$\ell(w^\top x_k) = \ell(w_t^\top x_k) + \partial\ell(w_t^\top x_k)(w^\top x_k - w_t^\top x_k) + \frac{\partial^2\ell(z)}{2}(w^\top x_k - w_t^\top x_k)^2$$

for some  $z = \tilde{w}_t^\top x$  where  $\|\tilde{w}_t - w_t\| \leq R_t$  since our feasible region satisfies  $\|w - w_t\| \leq R_t$ . The key step is to bound the term  $\partial^2\ell(z)$ . This bound will depend on the structure of the loss function. We now derive these bounds for logistic regression. We want the following to ensure that the condition in Theorem 2 with  $l_t = \sum_{x_k \in G_t} \partial\ell(w_t^\top x_k)x_k$  and  $c = \sum_{x_k \in G_t} [\ell(w_t^\top x_k) - \partial\ell(w_t^\top x_k)w_t^\top x_k]$ :

$$\frac{\partial^2\ell(z)}{2}(w^\top x_k - w_t^\top x_k)^2 \leq \delta$$

for all  $x_k \in G_t$ . The above statement is true when

$$\frac{\partial^2\ell(z)}{2}R_t^2\|x_k\|^2 \leq \delta \quad (12)$$

This can be obtained by a straightforward application of the Cauchy-Schwartz inequality. The final step is to derive an upper bound on  $\partial^2\ell(z)$ . For logistic loss we have

$$\partial^2\ell(z) = \frac{1}{(1 + \exp(-z))(1 + \exp(z))}.$$

Without loss of generality, we can assume  $z > 0$ . Then we have  $\partial^2\ell(z) \leq 1/(1 + \exp(z))$ . Using the above inequality it is easy to see that Equation (12) is satisfied if

$$\frac{R_t^2\|x_k\|^2}{2(1 + \exp(z))} \leq \delta.$$

We observe that

$$\begin{aligned} \frac{R_t^2\|x_k\|^2}{2(1 + \exp(z))} &= \frac{R_t^2\|x_k\|^2}{2(1 + \exp(w_t^\top x_k + z - w_t^\top x_k))} \\ &\leq \frac{R_t^2\|x_k\|^2}{2(1 + \exp(w_t^\top x_k) \exp(-R_t\|x_k\|))} \end{aligned}$$

This follows from the fact that  $z = \tilde{w}_t^\top x$  where  $\|\tilde{w}_t - w_t\| \leq R_t$ . Hence, for logistic regression, the goal of LINEARAPPROX is to identify all points satisfying

$$V_t(x_k) = \frac{R_t^2\|x_k\|^2}{2(1 + \exp(w_t^\top x_k) \exp(-R_t\|x_k\|))} \leq \delta$$

and place these points in the set  $G_t$ . The linear function to be used for approximation is obtained from the first-order Taylor expansion. The pseudocode for LINEARAPPROX in case of logistic regression is given in Algorithm 3.

---

**Algorithm 3** LINEARAPPROX for Logistic Regression

---

**INPUT:**  $P', w_t, R_t$

- 1:  $G_t = \{x_k \in P' \mid V_t(x_k) \leq \delta\}$
  - 2:  $l_t = -\sum_{x_k \in G_t} \frac{x_k}{(1 + \exp(w_t^\top x_k))}$
- 

Furthermore, we can also obtain a relationship between the margin of  $x_k$  with respect to the optimal solution and the iteration at which the point  $x_k$  moves to the set  $G_t$ . We note the following:

$$\begin{aligned} \frac{R_t^2\|x_k\|^2}{2(1 + \exp(z))} &= \frac{R_t^2\|x_k\|^2}{2(1 + \exp(w_*^\top x_k + z - w_*^\top x_k))} \\ &\leq \frac{R_t^2\|x_k\|^2}{2(1 + \exp(M_k^* \exp(-2R_t\|x_k\|))} \\ &\leq \frac{R_t^2\|x_k\|^2 \exp(2R_t\|x_k\|)}{2 \exp(M_k^*)} \leq \frac{\exp(3R_t\|x_k\|)}{2 \exp(M_k^*)} \end{aligned}$$

where  $M_k^* = |w_*^\top x_k|$ . The first step follows from triangle inequality and the fact that  $z = \tilde{w}_t^\top x$  where  $\|\tilde{w}_t - w_t\| \leq R_t$  and  $\|\tilde{w}_t - w_t\| \leq R_t$ . The final step follows from the fact that  $x^2 \leq \exp(x)$  for  $x \geq 0$ . Therefore, from above inequality it is easy to see that Equation (12) is satisfied when the following holds

$$R_t \leq \max \left\{ \frac{\sqrt{\delta}}{\|x_k\|}, \frac{M_k^* + \log(2\delta)}{3\|x_k\|} \right\}. \quad (13)$$

**LINEARAPPROX for SVM**

For SVM, the implementation of LINEARAPPROX is pretty straightforward. Due to the piecewise linear nature of the hinge loss, the condition in Equation (10) is satisfied with  $\delta = 0$  if  $w^\top x_k$  is greater than 1 (or less than 1) for the whole feasible region  $\|w - w_t\| \leq R_t$ . This is satisfied when

$$R_t \leq \frac{|1 - w_t^\top x_k|}{\|x_k\|}$$

The pseudocode for LINEARAPPROX in case of SVM is given in Algorithm 4.

---

**Algorithm 4** LINEARAPPROX for SVM

---

**INPUT:**  $P', w_t, R_t$

- 1:  $G_t = \{x_k \in P' \mid R_t \leq |1 - w_t^\top x_k|/\|x_k\|\}$
  - 2:  $l_t = -\sum_{x_k \in G_t} \mathbb{I}(w_t^\top x_k < 1)x_k$
- 

Similarly to the case of logistic regression, we analyze how the margin of  $x_k$  affects when it get included in the set  $G_t$ . For this, note the following:

$$1 - w_t^\top x_k = 1 - w_*^\top x_k - (w_t - w_*)^\top x_k$$



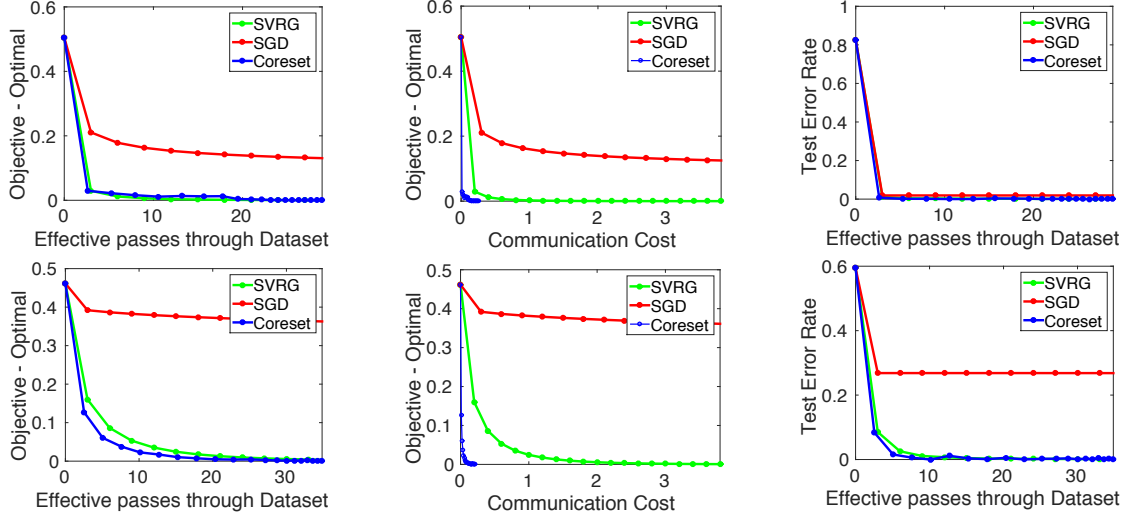


Figure 1:  $l_2$ -regularized logistic regression on ijcnn1 (top) and cod-rna (bottom) datasets. We compare our algorithm with mini-batch SVRG and SGD. Training loss residual is shown with respect to passes through the dataset and communication cost (left and central columns). Test error with respect to the passes through the dataset is shown in the right column.

Again, the above quantity will not change sign when  $w^\top x_k$  is greater than 1 (or less than 1) for the whole feasible region  $\|w - w_t\| \leq R_t$ . Based on the expression above, this is satisfied when

$$\|w_t - w_*\| \|x_k\| \leq |1 - w_*^\top x_k| = M_k^*$$

It is obtained by application of the Cauchy-Schwartz inequality. Note the difference in the definition of  $M_k^*$  in comparison to logistic regression. Hence, condition in Equation (10) will be satisfied when

$$R_t \leq \frac{M_k^*}{\|x_k\|} \quad (14)$$

Let us make a final remark before proceeding to the experimental section. It should be emphasized that based on Equations (13) and (14), the cardinality of  $G_t$  critically depends on the margin of the training points. If the margin of the training points is large, then the coreset size is small and consequently the communication and computation costs are low. Hence, our algorithm is naturally adaptive to the hardness of the optimization problem.

## 5 EXPERIMENTS

We present our empirical results in this section. To evaluate the performance of our algorithm, we focus on the task of regularized logistic regression. Recall that Equation (1) in this case is of the following form:

$$\min_w f(w) \equiv \frac{\lambda}{2} \|w\|^2 + \frac{1}{n} \sum_{i=1}^n \log(1 + \exp(-y_i w^\top x_i)).$$

In our Matlab implementation, we measure simulated communication costs. We use the following datasets.

| Dataset | # examples | # features |
|---------|------------|------------|
| ijcnn1  | 49,990     | 22         |
| cod-rna | 59,535     | 8          |
| w8a     | 64,700     | 300        |
| covtype | 581,012    | 54         |

All these datasets can be accessed from the LIBSVM website.<sup>3</sup> Similarly to [7], each dataset is scaled to  $[-1, 1]$ . We split each dataset in 3:1 ratio for training and testing purposes respectively.

The regularization parameter  $\lambda$  in Equation (1) is  $1/n$ . Recall that  $n$  is the size of the training set. This results in a high condition number and consequently increases the difficulty of the problem [10]. All the experiments were conducted for 10 random seeds and results are reported by averaging over these 10 runs.

We use PROXSVRG [16] for solving the subproblems of Algorithm 2 at each iteration. SVRG is an incremental first-order method that can be used for solving optimization problems conforming to Equation (1). The origin is used as the initial point for the for all our experiments. The number of “inner iterations” in the PROXSVRG algorithm is set to the recommended value of  $m = 2n$ . The step size parameter for each dataset is chosen so as to give the fastest convergence for PROXSVRG.

For our experiments, we choose  $T = 20$  and  $\gamma_1 = \gamma_2 = \dots = \gamma_T = \gamma$  in Algorithm 2 where  $\gamma$  is chosen such that upper bound on  $R_T$  is 0.01. Such a choice is rea-

<sup>3</sup><http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/>

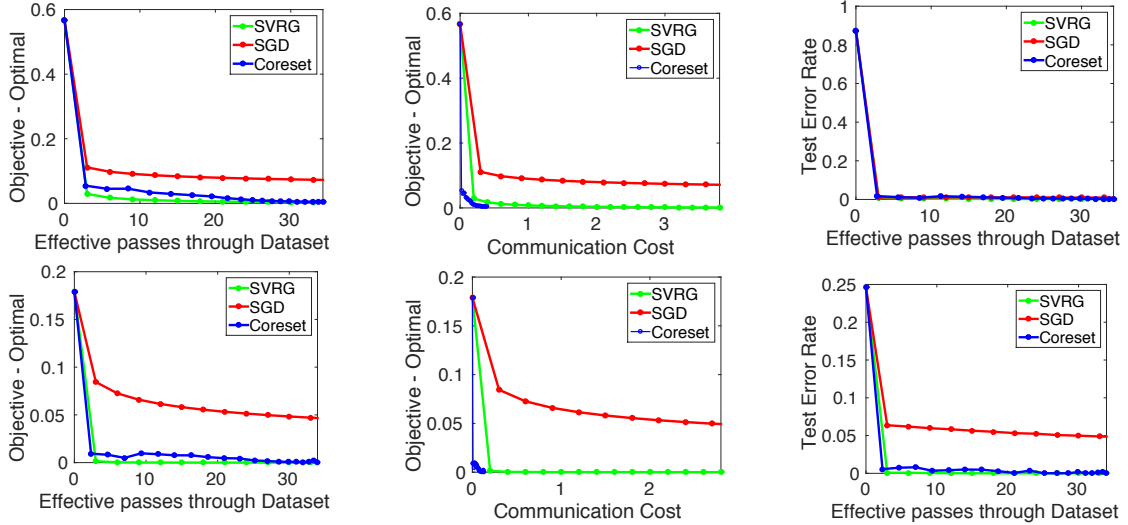


Figure 2:  $l_2$ -regularized logistic regression on more datasets w8a (top) and coverype (bottom). Similar to the previous case, we compare our algorithm with mini-batch SVRG and SGD.

sonable in the case of linearly convergent algorithms — which is the scenario we anticipate for our algorithm. As mentioned earlier, we use a clustering based algorithm instead of the cover. The main rationale behind such a choice is the availability of coresets for data clustering problems. As a heuristic, we directly use  $k$ -means coresets for Algorithm 2. The sensitivity based coreset for  $k$ -means is used in all our experiments. We refer interested reader to [4, 5] for more details of the coreset. We set the coreset size to be 500 for ijcnn1 and cod-rna datasets. This value is set to 1000 and 3000 for w8a and coverype datasets respectively. Note that these coreset sizes are much smaller in comparison to the training data.

We compare our algorithm with SVRG [7] and SGD. A mini-batch version of these methods is used in order to reduce the communication cost of these approaches. We use a mini-batch size  $b = 10$  in all our experiments. The number of inner iterations in SVRG is  $m = \lceil \frac{2n}{b} \rceil$  in all our experiments in order to limit the total inner iterations to the recommended  $2n$  iterations. For SGD, we use the learning rate of  $\alpha/\sqrt{t}$  where  $\alpha$  is the step size used for the all the algorithms for that dataset.

We report the training loss residual i.e., objective value in Equation (1) achieved by the algorithms minus the optimal objective value (obtained by running gradient descent for a very long time) and the test error rate of the algorithms with respect to the number of effective passes through the dataset. This includes the cost for calculating the gradients and the coresets. This provides information about the computation complexity of the algorithm. To measure the communication cost of the algorithm, we use the ratio of the number of  $d$  dimensional vectors communicated to the size of the training data.

Figures 1 and 2 show the performance of the algorithms on the aforementioned datasets. We have several observations from these empirical results. First, we observe that SVRG outperforms SGD in terms of all the metrics of our interest. This observation is not surprising given the linear convergence of SVRG in comparison to the sub-linear convergence of SGD (see [7] for more details). We then observe that our algorithm is competitive to SVRG in terms of training loss residual and test error rate (shown in the first and the third columns of the figures respectively). However, our major gain is in the communication cost of the algorithm. As seen in these figures, our algorithm performs much better in comparison to other algorithms in terms of communication cost. In other words, for the same communication cost, our algorithm has a much lower objective value when compared to SVRG and SGD. We believe that the performance of our algorithm can be further improved by utilizing the coresets of the previous iteration and is a part of our ongoing investigation. For future work, it will be interesting to test the performance of the algorithm on a real distributed environment.

## 6 CONCLUSION

This paper introduces a novel general strategy for designing communication efficient empirical loss minimization algorithms. The key to our approach is the concept of coresets — the idea of constructing a small summary of the training data and optimizing over this summary. We illustrated this strategy on two popular supervised learning problems — logistic regression and support vector machines. We presented convergence analysis for our algorithm. Furthermore, preliminary experiments show encouraging results in terms of both computational and communication costs.

## References

- [1] Pankaj K. Agarwal, Sariel Har-Peled, and Kasturi R. Varadarajan. Geometric approximation via coresets. In *Combinatorial and Computational Geometry, MSRI*, pages 1–30. University Press, 2005.
- [2] Maria-Florina Balcan, Steven Ehrlich, and Yingyu Liang. Distributed k-means and k-median clustering on general communication topologies. In Christopher J. C. Burges, Lon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *NIPS*, pages 1995–2003, 2013.
- [3] Stephen Boyd. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2010.
- [4] Dan Feldman and Michael Langberg. A unified framework for approximating and clustering data. In *Proceedings of the Forty-third Annual ACM Symposium on Theory of Computing, STOC '11*, pages 569–578, New York, NY, USA, 2011. ACM.
- [5] Dan Feldman, Melanie Schmidt, and Christian Sohler. Turning big data into tiny data: Constant-size coresets for k-means, pca and projective clustering. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '13*, pages 1434–1453. SIAM, 2013.
- [6] Thorsten Joachims. Training linear svms in linear time. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '06*, pages 217–226, New York, NY, USA, 2006. ACM.
- [7] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 315–323. Curran Associates, Inc., 2013.
- [8] Mu Li, Tong Zhang, Yuqiang Chen, and Alexander J. Smola. Efficient mini-batch training for stochastic optimization. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14*, pages 661–670, New York, NY, USA, 2014. ACM.
- [9] Dhruv Mahajan, S. Sathiya Keerthi, S. Sundararajan, and Léon Bottou. A functional approximation based distributed learning algorithm. *CoRR*, abs/1310.8418, 2013.
- [10] Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Mathematics and its applications. Kluwer Academic Publishers, 2004.
- [11] J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer series in operations research and financial engineering. Springer, 1999.
- [12] Katya Scheinberg. An efficient implementation of an active set method for svms. *Journal of Machine Learning Research*, 7:2237–2257, December 2006.
- [13] Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing finite sums with the stochastic average gradient. Technical report, 2013.
- [14] Shai Shalev-Shwartz, Yoram Singer, and Nathan Srebro. Pegasos: Primal estimated sub-gradient solver for svm. In *Proceedings of the 24th International Conference on Machine Learning, ICML '07*, pages 807–814, New York, NY, USA, 2007. ACM.
- [15] Ivor W. Tsang, James T. Kwok, Pak ming Cheung, and Nello Cristianini. Core vector machines: Fast svm training on very large data sets. *Journal of Machine Learning Research*, 6:363–392, 2005.
- [16] Lin Xiao and Tong Zhang. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075, jan 2014.
- [17] Martin A. Zinkevich, Alex Smola, Markus Weimer, and Lihong Li. Parallelized stochastic gradient descent. In *Advances in Neural Information Processing Systems 23*, pages 2595–2603, 2010.

---

# Large-scale randomized-coordinate descent methods with non-separable linear constraints

---

Sashank J. Reddi\* Ahmed Hefny\* Carlton Downey Avinava Dubey  
Machine Learning Department  
Carnegie Mellon University

Suvrit Sra†  
Massachusetts Institute of Technology

## Abstract

We develop randomized block coordinate descent (CD) methods for linearly constrained convex optimization. Unlike other large-scale CD methods, we do not assume the constraints to be separable, but allow them be coupled linearly. To our knowledge, ours is the first CD method that allows linear coupling constraints, without making the global iteration complexity have an *exponential dependence* on the number of constraints. We present algorithms and theoretical analysis for four key (convex) scenarios: (i) smooth; (ii) smooth + separable nonsmooth; (iii) asynchronous parallel; and (iv) stochastic. We discuss some architectural details of our methods and present preliminary results to illustrate the behavior of our algorithms.

## 1 INTRODUCTION

Coordinate descent (CD) methods are conceptually among the simplest schemes for unconstrained optimization—they have been studied for a long time (see e.g., [1, 4, 28]), and are now enjoying greatly renewed interest. Their resurgence is rooted in successful applications in machine learning [15, 16], statistics [8, 17], and many other areas—see [31, 32, 35] and references therein for more examples.

A catalyst to the theoretical as well as practical success of CD methods has been randomization. (The idea of randomized algorithms for optimization methods is of course much older, see e.g., [29].) Indeed, generic non-randomized CD has resisted complexity analysis, though there is promising recent work [14, 34, 40]; remarkably for randomized CD for smooth convex optimization, Nesterov [25, 26] presented an analysis of global iteration complexity. This work triggered several improvements, such as

[32, 33], who simplified and extended the analysis to include separable nonsmooth terms. Randomization has also been crucial to a host of other CD algorithms and analyses [5, 16, 20, 23, 30, 31, 33, 35–37].

Almost all of the aforementioned CD methods assume essentially unconstrained problems, which at best allow separable constraints. In contrast, we develop, analyze, and implement randomized CD methods for the following composite objective convex problem with *non-separable* linear constraints

$$\min_x F(x) := f(x) + h(x) \quad \text{s.t.} \quad Ax = 0. \quad (1)$$

Here,  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  is assumed to be continuously differentiable and convex, while  $h : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{\infty\}$  is lower semi-continuous, convex, coordinate-wise separable, but not necessarily smooth; the linear constraints (LC) are specified by a matrix  $A \in \mathbb{R}^{m \times n}$ , for which  $m \ll n$ , and a certain structure (see §4) is assumed. The reader may wonder whether one cannot simply rewrite Problem (1) in the form of  $f + h$  (without additional constraints) using suitable indicator functions. However, the resulting regularized problem then *no longer* fits the known efficient CD frameworks [32], since the nonsmooth part is *not* block-separable.

Problem (1) subsumes the usual regularized optimization problems pervasive in machine learning for the simplest ( $m = 0$ ) case. In the presence of linear constraints ( $m > 0$ ), Problem (1) assumes a form used in the classic Alternating Direction Method of Multipliers (ADMM) [9, 10]. The principal difference between our approach and ADMM is that the latter treats the entire variable  $x \in \mathbb{R}^n$  as a single block, whereas we use the structure of  $A$  to split  $x$  into  $b$  smaller blocks. Familiar special cases of Problem (1) include SVM (with bias) dual, fused Lasso and group Lasso [38], and linearly constrained least-squares regression [11, 19].

Recently, Necoara et al. [23] studied a special case of Problem (1) that sets  $h \equiv 0$  and assumes a single sum constraint. They presented a randomized CD method that starts with a feasible solution and at each iteration updates a pair

\*Indicates equal contribution.

† A part of this work was performed when the author was at Carnegie Mellon University

of coordinates to ensure descent on the objective while maintaining feasibility. This scheme is reminiscent of the well-known SMO procedure for SVM optimization [27]. For smooth convex problems with  $n$  variables, Necoara et al. [23] prove an  $O(1/\epsilon)$  rate of convergence. More recently, in [22] considered a generalization to the general case  $Ax = 0$  (assuming  $h$  is coordinatewise separable).

Unfortunately, the analysis in [22] yields an extremely pessimistic complexity result:

**Theorem 1** ([22]). *Consider Problem (1) with  $h$  being coordinatewise separable, and  $A \in \mathbb{R}^{m \times n}$  with  $b$  blocks. Then, the CD algorithm in [22] takes no more than  $O(b^m/\epsilon)$  iterations to obtain a solution of  $\epsilon$ -accuracy.*

This result is exponential in the number of constraints and too severe even for small-scale problems!

We present randomized CD methods, and prove that for important special cases (mainly  $h \equiv 0$  or  $A$  is a sum constraint) we can obtain global iteration complexity that *does not have* an intractable dependence on either the number of coordinate blocks ( $b$ ), or on the number of linear constraints ( $m$ ). Previously, Tseng and Yun [39] also studied a linearly coupled block-CD method based on the Gauss-Southwell choice; however, their complexity analysis applies only to the special  $m = 0$  and  $m = 1$  cases.

To our knowledge, ours is the first work on CD for problems with more than one ( $m > 1$ ) linear constraints that presents such results.

**Contributions.** In light of the above background, the primary contributions of this paper are as follows:

- Convergence rate analysis of a randomized block-CD method for the smooth case ( $h \equiv 0$ ) with  $m \geq 1$  general linear constraints.
- A tighter convergence analysis for the composite function optimization ( $h \neq 0$ ) than [22] in the case of sum constraint.
- An asynchronous CD algorithm for Problem (1).
- A stochastic CD method with convergence analysis for solving problems with a separable loss  $f(x) = (1/N) \sum_{i=1}^N f_i(x)$ .

Table 1 summarizes our contributions and compares it with existing state-of-the-art coordinate descent methods. The detailed proofs of all our theoretical claims are available in the appendix.

**Additional related work.** As noted, CD methods have a long history in optimization and they have gained tremendous recent interest. We cannot hope to do full justice to all the related work, but refer the reader to [32, 33] and [20] for more thorough coverage. Classically, local linear convergence was analyzed in [21]. Global rates for randomized

| Paper | LC         | Prox       | Parallel   | Stochastic |
|-------|------------|------------|------------|------------|
| [23]  | YES        | ×          | ×          | ×          |
| [39]  | YES        | YES        | ×          | ×          |
| [22]  | YES        | YES        | ×          | ×          |
| [7]   | ×          | YES        | YES        | ×          |
| [5]   | ×          | $\ell_1$   | YES        | YES        |
| [33]  | ×          | YES        | YES        | ×          |
| Ours  | <b>YES</b> | <b>YES</b> | <b>YES</b> | <b>YES</b> |

Table 1: Summary comparison of our method with other CD methods; LC denotes ‘linear constraints’; Prox signifies an extension using proximal operators (to handle  $h \neq 0$ ).

block coordinate descent (BCD) were pioneered by Nesterov [25], and have since then been extended by various authors [2, 32, 33, 40]. The related family of Gauss-Seidel like analyses for ADMM have also recently gained prominence [13]. A combination of randomized block-coordinate ideas with Frank-Wolfe methods was recently presented in [18], though algorithmically the Frank-Wolfe approach is very different as it relies on non projection based oracles.

## 2 PRELIMINARIES

In this section, we further explain our model and assumptions. We assume that the entire space  $\mathbb{R}^n$  is decomposed into  $b$  blocks, i.e.,  $x = [x_1^\top, \dots, x_b^\top]^\top$  where  $x \in \mathbb{R}^n$ ,  $x_i \in \mathbb{R}^{n_i}$  for all  $i \in [b]$ , and  $n = \sum_i n_i$ . For any  $x \in \mathbb{R}^n$ , we use  $x_i$  to denote the  $i^{\text{th}}$  block of  $x$ . We model communication constraints in our algorithms by viewing variables as nodes in a connected graph  $G := (V, E)$ . Specifically, node  $i \in V \equiv [b]$  corresponds to variable  $x_i$ , while an edge  $(i, j) \in E \subset V \times V$  is present if nodes  $i$  and  $j$  can exchange information. We use ‘pair’ and ‘edge’ interchangeably.

For a differentiable function  $f$ , we use  $f_{i_1 \dots i_p}$  and  $\nabla_{i_1 \dots i_p} f(x)$  (or  $\nabla_{x_{i_1} \dots x_{i_p}} f(x)$ ) to denote the restriction of the function and its partial gradient to coordinate blocks  $(x_{i_1}, \dots, x_{i_p})$ . For any matrix  $B$  with  $n$  columns, we use  $B_i$  to denote the columns of  $B$  corresponding to  $x_i$  and  $B_{ij}$  to denote the columns of  $B$  corresponding to  $x_i$  and  $x_j$ . We use  $U$  to denote the  $n \times n$  identity matrix and hence  $U_i$  is a matrix that places an  $n_i$  dimensional vector into the corresponding block of an  $n$  dimensional vector.

We make the following standard assumption on the partial gradients of  $f$ .

**Assumption 1.** The function has block-coordinate Lipschitz continuous gradient, i.e.,

$$\|\nabla_i f(x) - \nabla_i f(x + U_i h)\| \leq L_i \|h_i\| \text{ for all } x \in \mathbb{R}^n, .$$

Assumption 1 is similar to the typical Lipschitz continuous gradients assumed in first-order methods and it is necessary to ensure convergence of block-coordinate methods.

When functions  $f_i$  and  $f_j$  have Lipschitz continuous gradients with constants  $L_i$  and  $L_j$  respectively, one can show that the function  $f_{ij}$  has a Lipschitz continuous gradient with  $L_{ij} = L_i + L_j$  [22; Lemma 1]. The following result is standard.

**Lemma 2.** For any function  $g : \mathbb{R}^n \rightarrow \mathbb{R}$  with  $L$ -Lipschitz continuous gradient  $\nabla g$ , we have

$$g(x) \leq g(y) + \langle \nabla g(y), x - y \rangle + \frac{L}{2} \|x - y\|^2, \quad x, y \in \mathbb{R}^n.$$

Following [32, 39], we also make the following assumption on the structure of  $h$ .

**Assumption 2.** The nonsmooth function  $h$  is block separable, i.e.,  $h(x) = \sum_i h_i(x_i)$ .

This assumption is critical to composite optimization using CD methods. We also assume access to an oracle that returns function values and *partial gradients* at any points and iterates of the optimization algorithm.

### 3 ALGORITHM

We are now ready to present our randomized CD methods for Problem (1) in various settings. We first study composite minimization (§3.1) and later look at asynchronous (§3.2) and stochastic (§3.3) variants. The main idea underlying our algorithms is to pick a random pair  $(i, j) \in E$  of variables (blocks) at each iteration, and to update them in a manner which maintains feasibility and ensures progress in optimization.

#### 3.1 Composite Minimization

We begin with the nonsmooth setting, where  $h \neq 0$ . We start with a feasible point  $x^0$ . Then, at each iteration we pick a random pair  $(i, j) \in E$  of variables and minimize the first-order Taylor expansion of the loss  $f$  around the current iterate while maintaining feasibility. Formally, this involves performing the update

$$Z(f, x, (i, j), \alpha) := \arg \min_{A_{ij}d_{ij}=0} f(x) + \langle \nabla_{ij} f(x), d_{ij} \rangle + (2\alpha)^{-1} \|d_{ij}\|^2 + h(x + U_{ij}d_{ij}), \quad (2)$$

where  $\alpha > 0$  is a stepsize parameter and  $d_{ij}$  is the update. The right hand side of Equation (2) upper bounds  $f$  at  $x + U_{ij}d_{ij}$ , as seen by using Assumption 1 and Lemma 2. If  $h(x) \equiv 0$ , minimizing Equation (2) yields

$$\begin{aligned} \lambda &\leftarrow \alpha(A_i A_i^\top + A_j A_j^\top)^+ (A_i \nabla_i f(x) + A_j \nabla_j f(x)) \\ d_i &\leftarrow -\alpha \nabla_i f(x) + A_i^\top \lambda \\ d_j &\leftarrow -\alpha \nabla_j f(x) + A_j^\top \lambda \end{aligned} \quad (3)$$

Algorithm 1 presents the resulting method.

Note that since we start with a feasible point  $x^0$  and the update  $d^k$  satisfies  $Ad^k = 0$ , the iterate  $x^k$  is always feasible. However, it can be shown that a necessary condition for Equation (2) to result in a non-zero update is that

$A_i$  and  $A_j$  span the same column space. If the constraints are not block separable (i.e. for any partitioning of blocks  $x_1, \dots, x_b$  into two groups, there is a constraint that involves blocks from both groups), a typical way to satisfy the aforementioned condition is to require  $A_i$  to be full row-rank for all  $i \in [b]$ . This constraints the minimum block size to be chosen in order to apply randomized CD.

Theorem 3 describes convergence of Algorithm 1 for the smooth case ( $h \equiv 0$ ), while Theorem 6 considers the nonsmooth case under a suitable assumption on the structure of the interdependency graph  $G$ —both results are presented in Section 4.

```

1:  $x^0 \in \mathbb{R}^n$  such that  $Ax^0 = 0$ 
2: for  $k \geq 0$  do
3:   Select a random edge  $(i_k, j_k) \in E$  with probability
      $P^{i_k j_k}$ 
4:    $d^k \leftarrow U_{i_k j_k} Z(f, x^k, (i_k, j_k), \alpha_k / L_{i_k j_k})$ 
5:    $x^{k+1} \leftarrow x^k + d^k$ 
6:    $k \leftarrow k + 1$ 
7: end for

```

**Algorithm 1:** Composite Minimization with Linear Constraints

#### 3.2 Asynchronous Parallel Algorithm for Smooth Minimization

Although the algorithm described in the previous section solves a simple subproblem at each iteration, it is inherently sequential. This can be a disadvantage when addressing large-scale problems. To overcome this concern, we develop an asynchronous parallel method that solves Problem (1) for the smooth case.

Our parallel algorithm is similar to Algorithm 1, except for a crucial difference: now we may have multiple processors, and each of these executes the loop 2–6 *independently* without the need for coordination. This way, we can solve subproblems (i.e., multiple pairs) simultaneously in parallel, and due to the asynchronous nature of our algorithm, we can execute updates as they complete, without requiring any locking.

The critical issue, however, with implementing an asynchronous algorithm in the presence of non-separable constraints is ensuring feasibility throughout the course of the algorithm. This requires the operation  $x_i \leftarrow x_i + \delta$  to be executed in an *atomic* (i.e., sequentially consistent) fashion. Modern processors facilitate that without an additional locking structure through the “compare-and-swap” instruction [30]. Since the updates use atomic increments and each update satisfies  $Ad^k = 0$ , the net effect of  $T$  updates is  $\sum_{k=1}^T Ad^k = 0$ , which is feasible despite asynchronicity of the algorithm.

The next key issue is that of convergence. In an asynchronous setting, the updates are based on *stale* gradients

that are computed using values of  $x$  read many iterations earlier. But provided that gradient staleness is bounded, we can establish a sublinear convergence rate of the asynchronous parallel algorithm (Theorem 4). More formally, we assume that in iteration  $k$ , *stale* gradients are computed based on  $x^{D(k)}$  such that  $k - D(k) \leq \tau$ . The bound on staleness, denoted by  $\tau$ , captures the degree of parallelism in the method: such parameters are typical in asynchronous systems and provides a bound on the delay of the updates [20].

Before concluding the discussion on our asynchronous algorithm, it is important to note the difficulty of extending our algorithm to nonsmooth problems. For example, consider the case where  $h = \mathbb{I}_C$  (indicator function of some convex set). Although a pairwise update as suggested above maintains feasibility with respect to the linear constraint  $Ax = 0$ , it may violate the feasibility of being in the convex set  $C$ . This complication can be circumvented by using a convex combination of the current iterate with the update, as this would retain overall feasibility. However, it would complicate the convergence analysis. We plan to investigate this direction in future work.

### 3.3 Stochastic Minimization

An important subclass of Problem (1) assumes separable losses  $f(x) = \frac{1}{N} \sum_{i=1}^N f_i(x)$ . This class arises naturally in many machine learning applications where the loss separates over training examples. To take advantage of this added separability of  $f$ , we can derive a stochastic block-CD procedure.

Our key innovation here is the following: in addition to randomly picking an edge  $(i, j)$ , we also pick a function randomly from  $\{f_1, \dots, f_N\}$  and perform our update using this function. This choice substantially reduces the cost of each iteration when  $N$  is large, since now the gradient calculations involve only the randomly selected function  $f_i$  (i.e., we now use a stochastic-gradient). Pseudocode is given in Algorithm 2.

```

1: Choose  $x^0 \in \mathbb{R}^n$  such that  $Ax^0 = 0$ .
2: for  $k \geq 0$  do
3:   Select a random edge  $(i_k, j_k) \in E$  with probability
      $p_{i_k j_k}$ 
4:   Select random integer  $l \in [N]$ 
5:    $x^{k+1} \leftarrow x^k + U_{i_k j_k} Z(f_l, x^k, (i_k, j_k), \alpha_k / L_{i_k j_k})$ 
6:    $k \leftarrow k + 1$ 
7: end for

```

**Algorithm 2:** Stochastic Minimization with Linear Constraints

Notice that the per iteration cost of Algorithm 2 is lower than Algorithm 1 by a factor of  $N$ . However, as we will see later, this speedup comes at a price of slower convergence rate (Theorem 5). Moreover, to ensure convergence,

decaying step sizes  $\{\alpha_k\}_{k \geq 0}$  are generally chosen.

## 4 CONVERGENCE ANALYSIS

In this section, we outline convergence results for the algorithms described above. The proofs are somewhat technical, and hence left in the appendix due to lack of space; here we present only the key ideas.

For simplicity, we present our analysis for the following reformulation of the main problem:

$$\begin{aligned} \min_{y,z} \quad & f(y, z) + \sum_{i=1}^b h(y_i, z_i) \\ \text{subject to} \quad & \sum_{i=1}^b y_i = 0, \end{aligned} \quad (4)$$

where  $y_i \in \mathbb{R}^{n_y}$  and  $z_i \in \mathbb{R}^{n_z}$ . Let  $y = [y_1^\top \dots y_b^\top]^\top$  and  $z = [z_1^\top \dots z_b^\top]^\top$ . We use  $x$  to denote the concatenated vector  $[y^\top z^\top]^\top$  and hence we assume (unless otherwise mentioned) that the constraint matrix  $A$  is defined as follows

$$A \begin{pmatrix} y \\ z \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^b y_i \\ 0 \end{pmatrix}. \quad (5)$$

It is worth emphasizing that this analysis *does not* result in any loss of generality. This is due to the fact that Problem (1) with a general constraint matrix  $\tilde{A}$  having full row-rank submatrices  $\tilde{A}_i$ 's can be rewritten in the form of Problem (4) by using the transformation specified in Section E of the appendix. It is important to note that this reduction is presented only for the ease of exposition. For our experiments, we directly solve the problem in Equation 2.

Let  $\eta_k = \{(i_0, j_0), \dots, (i_{k-1}, j_{k-1})\}$  denote the pairs selected up to iteration  $k - 1$ . To simplify notation, assume (without loss of generality) that the Lipschitz constant for the partial gradient  $\nabla_i f(x)$  and  $\nabla_{ij} f(x)$  is  $L$  for all  $i \in [n]$  and  $(i, j) \in E$ .

Similar to [23], we introduce a Laplacian matrix  $\mathcal{L} \in \mathbb{R}^{b \times b}$  that represents the communication graph  $G$ . Since we also have unconstrained variables  $z_i$ , we introduce a diagonal matrix  $\mathcal{D} \in \mathbb{R}^{b \times b}$ .

$$\mathcal{L}_{ij} = \begin{cases} \sum_{r \neq i} \frac{p_{ir}}{2L} & i = j \\ -\frac{p_{ij}}{2L} & i \neq j \end{cases} \quad \mathcal{D}_{ij} = \begin{cases} \frac{p_i}{L} & i = j \\ 0 & i \neq j \end{cases}$$

We use  $\mathcal{K}$  to denote the concatenation of the Laplacian  $\mathcal{L}$  and the diagonal matrix  $\mathcal{D}$ . More formally,

$$\mathcal{K} = \begin{bmatrix} \mathcal{L} \otimes I_{n_y} & 0 \\ 0 & \mathcal{D} \otimes I_{n_z} \end{bmatrix}.$$

This matrix induces a norm  $\|x\|_{\mathcal{K}} = \sqrt{x^\top \mathcal{K} x}$  on the *feasible subspace*, with a corresponding dual norm

$$\|x\|_{\mathcal{K}}^* = \sqrt{x^\top \left( \begin{bmatrix} \mathcal{L}^+ \otimes I_{n_y} & 0 \\ 0 & \mathcal{D}^{-1} \otimes I_{n_z} \end{bmatrix} \right) x}$$

Let  $X^*$  denote the set of optimal solutions and let  $x^0$  denote the initial point. We define the following distance, which quantifies how far the initial point is from the optimal, taking into account the graph layout and edge selection probabilities

$$R(x^0) := \max_{x: f(x) \leq f(x^0)} \max_{x^* \in X^*} \|x - x^*\|_{\mathcal{K}} \quad (6)$$

**Note.** Before delving into the details of the convergence results, we would like to draw the reader's attention to the impact of the communication network  $G$  on convergence. In general, the convergence results depend on  $R(x^0)$ , which in turn depends on the Laplacian  $\mathcal{L}$  of the graph  $G$ . As a rule of thumb, the larger the connectivity of the graph, the smaller the value of  $R(x^0)$ , and hence, faster the convergence.

#### 4.1 Convergence results for the smooth case

We first consider the case when  $h = 0$ . Here the subproblem at  $k^{\text{th}}$  iteration has a very simple update  $d_{i_k j_k}^k = U_{i_k} d^k - U_{j_k} d^k$  where  $d^k = \frac{\alpha_k}{2L} (\nabla_{j_k} f(x^k) - \nabla_{i_k} f(x^k))$ . We now prove that Algorithm 1 attains an  $O(1/k)$  convergence rate.

**Theorem 3.** *Let  $\alpha_k = 1$  for  $k \geq 0$ , and let  $\{x^k\}_{k \geq 0}$  be the sequence generated by Algorithm 1; let  $f^*$  denote the optimal value. Then, we have the following rate of convergence:*

$$\mathbb{E}[f(x^k)] - f^* \leq \frac{2R^2(x^0)}{k}$$

where  $R(x^0)$  is as defined in Equation 6.

*Proof Sketch.* We first prove that each iteration leads to descent in expectation. More formally, we get

$$\mathbb{E}_{i_k j_k} [f(x^{k+1}) | \eta_k] \leq f(x^k) - \frac{1}{2} \nabla f(x^k)^\top \mathcal{K} \nabla f(x^k).$$

The above step can be proved using Lemma 2. Let  $\Delta_k = \mathbb{E}[f(x^k)] - f^*$ . It can be proved that

$$\frac{1}{\Delta_k} \leq \frac{1}{\Delta_{k+1}} - \frac{1}{2R^2(x^0)}$$

This follows from the fact that

$$\begin{aligned} f(x^{k+1}) - f^* &\leq \|x^k - x^*\|_{\mathcal{K}}^* \|\nabla f(x^k)\|_{\mathcal{K}} \\ &\leq R(x^0) \|\nabla f(x^k)\|_{\mathcal{K}} \quad \forall k \geq 0 \end{aligned}$$

Telescoping the sum, we get the desired result.  $\square$

Note that Theorem 3 is a strict generalization of the analysis in [23] and [22] due to: (i) the presence of unconstrained variables  $z$ ; and (ii) the presence of a non-decomposable objective function. It is also worth emphasizing that our

convergence rates improve upon those of [22], since they do not involve an exponential dependence of the form  $b^m$  on the number of constraints.

We now turn our attention towards the convergence analysis of our asynchronous algorithm under a consistent reading model [20]. In this context we would like to emphasize that while our theoretical analysis assumes consistent reads, we do not enforce this assumption in our experiments.

**Theorem 4.** *Let  $\rho > 1$  and  $\alpha_k = \alpha$  be such that  $\alpha < 2/(1 + \tau + \tau\rho^\tau)$  and  $\alpha < (\rho - 1)/(\sqrt{2}(\tau + 2)(\rho^{\tau+1} + \rho))$ . Let  $\{x_k\}_{k \geq 0}$  be the sequence generated by asynchronous algorithm using step size  $\alpha_k$  and let  $f^*$  denote the optimal value. Then, we have the following rate of convergence for the expected values of the objective function*

$$\mathbb{E}[f(x_k)] - f^* \leq \frac{R^2(x^0)}{\mu k}$$

where  $R(x^0)$  is as defined in Equation 6 and  $\mu = \frac{\alpha_k^2}{2} \left( \frac{1}{\alpha_k} - \frac{1 + \tau + \tau\rho^\tau}{2} \right)$ .

*Proof Sketch.* For ease of exposition, we describe the case where the unconstrained variables  $z$  are absent. The analysis of case with  $z$  variables can be carried out in a similar manner. Let  $D(k)$  denote the iterate of the variables used in the  $k^{\text{th}}$  iteration (the existence of  $D(k)$  follows from the consistent reading assumption). Let  $d^k = \frac{\alpha_k}{2L} (\nabla_{y_{j_k}} f(x^{D(k)}) - \nabla_{y_{i_k}} f(x^{D(k)}))$  and  $d_{i_k j_k}^k = x^{k+1} - x^k = U_{i_k} d^k - U_{j_k} d^k$ . Using Lemma 2 and the assumption that staleness in the variables is bounded by  $\tau$ , i.e.,  $k - D(k) \leq \tau$  and definition of  $d_{i_j}^k$ , we can derive the following bound:

$$\begin{aligned} \mathbb{E}[f(x^{k+1})] &\leq \mathbb{E}[f(x^k)] - L \left( \frac{1}{\alpha_k} - \frac{1 + \tau}{2} \right) \mathbb{E}[\|d_{i_k j_k}^k\|^2] \\ &\quad + \frac{L}{2} \mathbb{E} \left[ \sum_{t=1}^{\tau} \|d_{i_{k-t} j_{k-t}}^{k-t}\|^2 \right]. \end{aligned}$$

In order to obtain an upper bound on the norms of  $d_{i_k j_k}^k$ , we prove that

$$\mathbb{E} \left[ \|d_{i_{k-1} j_{k-1}}^{k-1}\|^2 \right] \leq \rho \mathbb{E} \left[ \|d_{i_k j_k}^k\|^2 \right]$$

This can be proven using mathematical induction. Using the above bound on  $\|d_{i_k j_k}^k\|^2$ , we get

$$\begin{aligned} \mathbb{E}[f(x^{k+1})] &\leq \\ \mathbb{E}[f(x^k)] &- L \left( \frac{1}{\alpha_k} - \frac{1 + \tau + \tau\rho^\tau}{2} \right) \mathbb{E}[\|d_{i_k j_k}^k\|^2] \end{aligned}$$

This proves that the method is a descent method in expectation. Following similar analysis as Theorem 3, we get the required result.  $\square$



Note the dependence of convergence rate on the staleness bound  $\tau$ . For larger values of  $\tau$ , the stepsize  $\alpha_k$  needs to be decreased to ensure convergence, which in turn slows down the convergence rate of the algorithm. Nevertheless, the convergence rate remains  $O(1/k)$ .

The last smooth case we analyze is our stochastic algorithm.

**Theorem 5.** *Let  $\alpha_i = \sqrt{\Delta_0 L} / (M\sqrt{i+1})$  for  $i \geq 0$  in Algorithm 2. Let  $\{x^k\}_{k \geq 0}$  be the sequence generated by Algorithm 2 and let  $f^*$  denote the optimal value. We denote  $\bar{x}^k = \arg \min_{0 \leq i \leq k} f(x^i)$ . Then, we have the following rate of convergence for the expected values of the objective:*

$$\mathbb{E}[f(\bar{x}^k)] - f^* \leq O\left(\frac{1}{\sqrt[4]{k}}\right)$$

where  $\Delta_0 = f(x^0) - f^*$ .

The convergence rate is  $O(1/k^{1/4})$  as opposed to  $O(1/k)$  of Theorem 3. On the other hand, the iteration complexity is lower by a factor of  $N$ ; this kind of tradeoff is typical in stochastic algorithms, where the slower rate is the price we pay for a lower iteration complexity. We believe that the convergence rate can be improved to  $O(1/\sqrt{k})$ , the rate generally observed in stochastic algorithms, by a more careful analysis.

## 4.2 Nonsmooth case

We finally state the convergence rate for the nonsmooth case ( $h \neq 0$ ) in the case of a sum constraint. Similar to [22], we assume  $h$  is coordinatewise separable (i.e. we can write  $h(x) = \sum_{i=1}^b \sum_j x_{ij}$ ), where  $x_{ij}$  is the  $j^{\text{th}}$  coordinate in the  $i^{\text{th}}$  block. For this analysis, we assume that the graph  $G$  is a clique<sup>1</sup> with uniform probability, i.e.,  $\lambda = p_{ij} = 2/b(b-1)$ .

**Theorem 6.** *Assume  $Ax = \sum_i A_i x_i$ . Let  $\{x^k\}_{k \geq 0}$  be the sequence generated by Algorithm 1 and let  $F^*$  denote the optimal value. Assume that the graph  $G$  is a clique with uniform probability. Then we have the following:*

$$\mathbb{E}[F(x^k) - F^*] \leq \frac{b^2 L R^2(x^0)}{2k + \frac{b^2 L R^2(x^0)}{\Delta_0}},$$

where  $R(x^0)$  is as defined in Equation 6.

This convergence rate is a generalization of the convergence rate obtained in Necoara and Patrascu [22] for a single linear constraint (see Theorem 1 in [22]). It is also an

<sup>1</sup>We believe our results also easily extend to the general case along the lines of [31–33], using the concept of *Expected Separable Overapproximation* (ESO). Moreover, the assumption is not totally impractical, e.g., in a multicore setting with a zero-sum constraint (i.e.  $A_i = I$ ), the clique-assumption introduces little cost.

improvement of the rate obtained in Necoara and Patrascu [22] for general linear constraints (see Theorem 4 in [22]) when applied to the special case of a sum constraint. Our improvement comes in the form of a tractable constant, as opposed to the exponential dependence  $O(b^m)$  shown in [22].

## 5 APPLICATIONS

To gain a better understanding of our approach, we state some applications of interest, while discussing details of Algorithm 1 and Algorithm 2 for them. While there are many applications of problem (1), due to lack of space we only mention a few prominent ones here.

**Support Vector Machines:** The SVM dual (with bias term) assumes the form (1); specifically,

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j z_i^\top z_j - \sum_{i=1}^n \alpha_i \\ \text{s.t.} \quad & \sum_i \alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C \quad \forall i \in [n]. \end{aligned} \quad (7)$$

Here,  $z_i$  denotes the feature vector of the  $i^{\text{th}}$  training example and  $y_i \in \{1, -1\}$  denotes the corresponding label. By letting  $f(\alpha) = \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j z_i^\top z_j - \sum_i \alpha_i$  and  $h(\alpha) = \sum_i \mathbb{I}(0 \leq \alpha_i \leq C)$  and  $A = [y_1, \dots, y_n]$  this problem can be written in form of Problem (1). Using Algorithm 1 for SVM involves solving a sub-problem similar to one used in SMO in the scalar case (i.e.,  $\alpha_i \in \mathbb{R}$ ) and can be solved in linear time in the block case (see [3]).

**Generalized Lasso:** The objective is to solve the following optimization problem.

$$\min_{\beta} \quad \frac{1}{2} \|Y - X\beta\|_2^2 + \lambda \|D\beta\|_1$$

where  $Y \in \mathbb{R}^N$  denotes the output,  $X \in \mathbb{R}^{N \times n}$  is the input and  $D \in \mathbb{R}^{q \times n}$  represents a specified penalty matrix. This problem can also be seen as a specific case of Problem (1) by introducing an auxiliary variable  $t$  and slack variables  $u, v$ . Then,  $f(\beta, t) = \frac{1}{2} \|Y - X\beta\|_2^2 + \sum_i t_i$ ,  $h(u, v) = \mathbb{I}(u \geq 0) + \mathbb{I}(v \geq 0)$  and,  $t - D\beta - u = 0$  and  $t + D\beta - v = 0$  are the linear constraints. To solve this problem, we can use either Algorithm 1 or Algorithm 2. In general, optimization of convex functions on a structured convex polytope can be solved in a similar manner.

**Unconstrained Separable Optimization:** Another interesting application is for unconstrained separable optimization. For any problem  $\min_x \sum_i f_i(x)$ —a form generally encountered across machine learning—can be rewritten using variable-splitting as  $\min_{\{x_i=x, \forall i \in [N]\}} \sum_i f_i(x_i)$ . Solving the problem in distributed environment requires considerable synchronization (for the consensus constraint), which can slow down the algorithm significantly. However, the dual of the problem is

$$\min_{\lambda} \sum_i f_i^*(\lambda_i) \quad \text{s.t.} \quad \sum_{i=1}^N \lambda_i = 0.$$

where  $f_i^*$  is the Fenchel conjugate of  $f_i$ . This reformulation perfectly fits our framework and can be solved in an asynchronous manner using the procedure described in Section 3.2.

Other interesting application include constrained least square problem, multi-agent planning problems, resource allocation—see [22, 23] and references therein for more examples.

## 6 EXPERIMENTS

In this section, we present our empirical results. In particular, we examine the behavior of random coordinate descent algorithms analyzed in this paper under different communication constraints and concurrency conditions.<sup>2</sup>

### 6.1 Effect of Communication Constraints

Our first set of experiments test the affect of the connectivity of the graph on the convergence rate. In particular, recall that the convergence analysis established in Theorem 3 depends on the Laplacian of the communication graph. In this experiment we demonstrate how communication constraints affect convergence in practice. We experiment with the following graph topologies of graph  $G$ : Ring, Clique, Star + Ring (i.e., the union of edges of a star and a ring) and Tree + Ring. On each layout we run the sequential Algorithm 1 on the following quadratic problem

$$\begin{aligned} \min \quad & C \sum_{i=1}^N \|x_i - (i \bmod 10)\mathbf{1}\|^2 \\ \text{s.t.} \quad & \sum_{i=1}^N A_i x_i = 0, \end{aligned} \quad (8)$$

Note the decomposable structure of the problem. For this experiment, we use  $N = 1000$  and  $x_i \in \mathbb{R}^{50}$ . We have 10 constraints whose coefficients are randomly generated from  $U[0, 1]$  and we choose  $C$  such that the objective evaluates to 1000 when  $x = 0$ .

The results for Algorithm 1 on each topology for 10000 iterations are shown in Figure 1. The results clearly show that better connectivity implies better convergence rate. Note that while the clique topology has significantly better convergence than other topologies, acceptable long-term performance can be achieved by much sparser topologies such as Star + Ring and Tree + Ring.

Having a sparse communication graph is important to lower the cost of a distributed system. Furthermore, it is worth mentioning that the sparsity of the communication graph is also important in a multicore setting; since Algorithm 1

<sup>2</sup>All experiments were conducted on a Google Compute Engine virtual machine of type “n1-highcpu-16”, which comprises 16 virtual CPUs and 14.4 GB of memory. For more details, please refer to <https://cloud.google.com/compute/docs/machine-types#highcpu>.

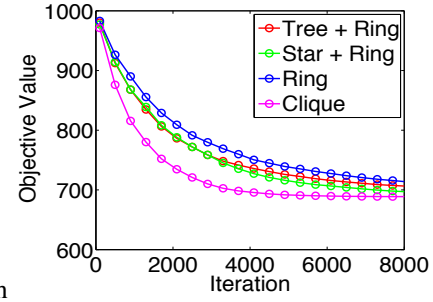


Figure 1: Objective value vs. number of iterations for different graph topologies. Note that larger the connectivity of the graph, faster is the convergence.

requires computing  $(A_i A_i^T + A_j A_j^T)^+$  for each communicating pair of nodes  $(i, j)$ . Our analysis shows that this computation takes a significant portion of the running time and hence it is essential to minimize the number of variable pairs that are allowed to be updated.

### 6.2 Concurrency and Synchronization

As seen earlier, compared to Tree + Ring, Star + Ring is a low diameter layout (diameter = 2). Hence, in a sequential setting, it indeed results in a faster convergence. However, Star + Ring requires a node to be connected to all other nodes. This high-degree node could be a contention point in a parallel setting. We test the performance of our asynchronous algorithm in this setting. To assess how the performance would be affected with such contention and how asynchronous updates would increase performance, we conduct another experiment on the synthetic problem (8) but on a larger scale ( $N = 10000$ ,  $x_i \in \mathbb{R}^{100}$ , 100 constraints).

Our concurrent update follows a master/slave scheme. Each thread performs a loop where in each iteration it elects a master  $i$  and slave  $j$  and then applies the following sequence of actions:

1. Obtain the information required for the update from the master (i.e., information for calculating the gradients used for solving the subproblem).
2. Send the master information to the slave, update the slave variable and get back the information needed to update the master.
3. Update the master based (only) on the information obtained from steps 1 and 2.

We emphasize that the master is not allowed to read its own state at step 3 except to apply an increment, which is computed based on steps 1 and 2. This ensures that the master’s increment is consistent with that of the slave, even if one or both of them was being concurrently overwritten by

another thread. More details on the implementation can be found in [12].

Given this update scheme, we experiment with three levels of synchronization: (a) **Double Locking:** Locks the master and the slave through the entire update. Because the objective function is decomposable, a more conservative locking (e.g. locking all nodes) is not needed. (b) **Single Locking:** Locks the master during steps 1 and 3 (the master is unlocked during step 2 and locks the slave during step 2). (c) **Lock-free:** No locks are used. Master and slave variables are updated through atomic increments similar to Hogwild! method.

Following [30], we use spinlocks instead of mutex locks to implement locking. Spinlocks are preferred over mutex locks when the resource is locked for a short period of time, which is the case in our algorithm. For each locking mechanism, we vary the number of threads from 1 to 15. We stop when  $f_0 - f_t > 0.99(f_0 - f^*)$ , where  $f^*$  is computed beforehand up to three significant digits. Similar to [30], we add artificial delay to steps 1 and 2 in the update scheme to model complicated gradient calculations and/or network latency in a distributed setting.

Figure 2 shows the speedup for Tree + Ring and Star + Ring layouts. The figure clearly shows that a fully synchronous method suffers from contention in the Star + Ring topology whereas asynchronous method does not suffer from this problem and hence, achieves higher speedups. Although the Tree + Ring layouts achieves higher speedup than Star + Ring, the latter topology results in much less running time ( $\sim 67$  seconds vs 91 seconds using 15 threads).

### 6.3 Practical Case Study: Parallel Training of Linear SVM

In this section, we explore the effect of parallelism on randomized CD for training a linear SVM based on the dual formulation stated in (7). Necoara et. al. [22] have shown that, in terms of CPU time, a sequential randomized CD outperforms coordinate descent using Gauss-Southwell selection rule. It was also observed that randomized CD outperforms LIBSVM [6] for large datasets while maintaining reasonable performance for small datasets.

In this experiment we use a clique layout. For SVM training in a multicore setting, using a clique layout does not introduce additional cost compared to a more sparse layout. To maintain the box constraint, we use the double-locking scheme described in Section 6.2 for updating a pair of dual variables.

One advantage of coordinate descent algorithms is that they do not require the storage of the Gram matrix; instead they can compute its elements on the fly. That comes, however, at the expense of CPU time. Similar to [22], to speed up gradient computations without increasing memory require-

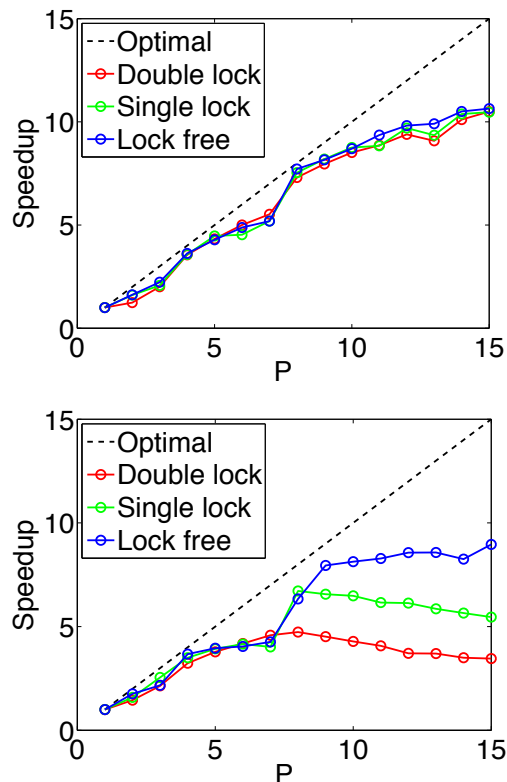


Figure 2: Speedup for Tree + Ring (top) and Star + Ring (bottom) topologies and different levels of synchronization. Note for Star + Ring topology, speedup of asynchronous algorithm is significantly higher than that of synchronous version.

ments, we maintain the primal weight vector of the linear SVM and use it to compute gradients. Basically, if we increment  $\alpha_i$  by  $\delta_i$  and  $\alpha_j$  by  $\delta_j$ , then we increment the weight vector by  $\delta_i y_i x_i + \delta_j y_j x_j$ . This increment is accomplished using atomic additions. However, this implies that all threads will be concurrently updating the primal weight vector. Similar to [30], we require these updates to be sparse with small overlap between non-zero coordinates in order to ensure convergence. In other words, we require training examples to have sparse features with small overlap between non-zero features.

We report speedups on two datasets used in [22].<sup>3</sup> Table 2 provides a description of both the datasets. For each dataset, we train the SVM model until  $f_0 - f_t > 0.9999(f_0 - f^*)$ , where  $f^*$  is the objective reported in [22]. In Figure 3, we report speedup for both the datasets. The figure shows that parallelism indeed increases the performance of randomized CD training of linear SVM.

<sup>3</sup>Datasets can be downloaded from <http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets>.

| Dataset | # of instances | # of features | Avg # of non-zero features |
|---------|----------------|---------------|----------------------------|
| a7a     | 16100          | 122           | 14                         |
| w8a     | 49749          | 300           | 12                         |

Table 2: Datasets used for linear SVM Speedup experiment

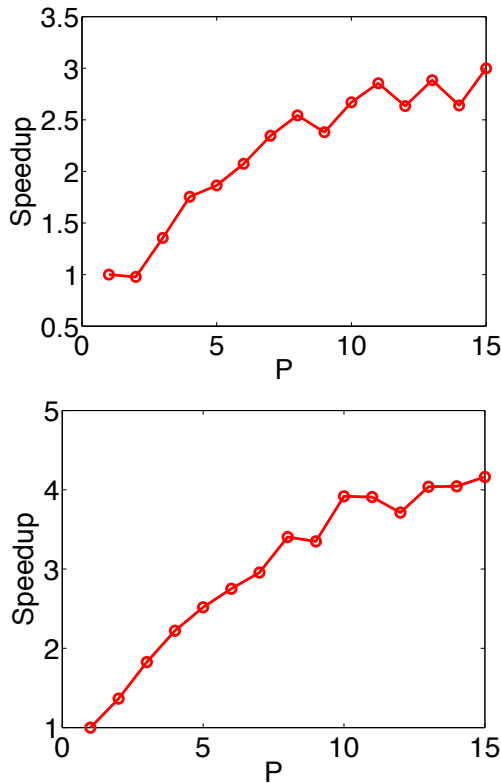


Figure 3: Speedup for linear SVM training on a7a (top) and w8a datasets.

## 7 DISCUSSION AND FUTURE WORK

We presented randomized coordinate descent methods for solving convex optimization problems with linear constraints that couple the variables. Moreover, we also presented composite objective, stochastic, and asynchronous versions of our basic method and provided their convergence analysis. We demonstrated the empirical performance of the algorithms. The experimental results of asynchronous algorithm look very promising.

There are interesting open problems for our problem in consideration: First, we would like to obtain high-probability results not just in expectation; another interesting direction is to extend the asynchronous algorithm to the non-smooth setting. Finally, while we obtain  $O(1/k)$  for general convex functions, obtaining an accelerated  $O(1/k^2)$  rate is a natural question.

## Acknowledgments

SS is partly supported by NSF grant: IIS-1409802. We thanks the anonymous reviewers for the helpful comments.

## References

- [1] A. Auslender. *Optimisation Méthodes Numériques*. Masson, 1976.
- [2] A. Beck and L. Tetruashvili. On the convergence of block coordinate descent type methods. *SIAM Journal on Optimization*, 23(4):2037–2060, 2013. doi: 10.1137/120887679.
- [3] P. Berman, N. Kover, and P. M. Pardalos. A linear-time algorithm for the least-distance problem. Technical report, Pennsylvania State University, Department of Computer Science, 1992.
- [4] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA, second edition, 1999.
- [5] J. Bradley, A. Kyrola, D. Bickson, and C. Guestrin. Parallel coordinate descent for L1-regularized loss minimization. In L. Getoor and T. Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning*, pages 321–328. Omnipress, 2011.
- [6] C.-C. Chang and C.-J. Lin. Libsvm: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2(3):27:1–27:27, May 2011. ISSN 2157-6904. doi: 10.1145/1961189.1961199. URL <http://doi.acm.org/10.1145/1961189.1961199>.
- [7] O. Fercoq and P. Richtárik. Accelerated, parallel and proximal coordinate descent. *CoRR*, abs/1312.5799, 2013.
- [8] J. Friedman, T. Hastie, H. Höfling, R. Tibshirani, et al. Pathwise coordinate optimization. *The Annals of Applied Statistics*, 1(2):302–332, 2007.
- [9] D. Gabay and B. Mercier. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers & Mathematics with Applications*, 2(1):17–40, 1976. doi: [http://dx.doi.org/10.1016/0898-1221\(76\)90003-1](http://dx.doi.org/10.1016/0898-1221(76)90003-1). URL <http://www.sciencedirect.com/science/article/pii/0898122176900031>.
- [10] R. Glowinski and A. Marrocco. Sur l’approximation, par éléments finis d’ordre un, et la résolution, par pénalisation-dualité d’une classe de problèmes de dirichlet non linéaires. *Revue Française d’Automatique, Informatique, et Recherche Opérationnelle*, 1975.
- [11] G. H. Golub and C. F. Van Loan. *Matrix Computations*. John Hopkins University Press, Baltimore, MD, 3rd edition, 1996.
- [12] A. Hefny, S. Reddi, and S. Sra. Coordinate descent algorithms with coupling constraints: Lessons learned. In *NIPS Workshop on Software Engineering For Machine Learning*, 2014.

- [13] M. Hong and Z.-Q. Luo. On the linear convergence of the alternating direction method of multipliers. *arXiv preprint arXiv:1208.3922*, 2012.
- [14] M. Hong, X. Wang, M. Razaviyayn, and Z.-Q. Luo. Iteration Complexity Analysis of Block Coordinate Descent Methods. *arXiv:1310.6957*, 2013.
- [15] C. J. Hsieh and I. S. Dhillon. Fast coordinate descent methods with variable selection for non-negative matrix factorization. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining(KDD)*, pages 1064–1072, August 2011.
- [16] C. J. Hsieh, K. W. Chang, C. J. Lin, S. S. Keerthi, and S. Sundararajan. A dual coordinate descent method for large-scale linear SVM. In W. Cohen, A. McCallum, and S. Roweis, editors, *ICML*, pages 408–415. ACM, 2008.
- [17] C.-J. Hsieh, M. A. Sustik, I. S. Dhillon, and P. D. Ravikumar. Sparse inverse covariance matrix estimation using quadratic approximation. In *NIPS*, pages 2330–2338, 2011.
- [18] S. Lacoste-Julien, M. Jaggi, M. Schmidt, and P. Pletscher. Block-coordinate frank-wolfe optimization for structural svms. *arXiv preprint arXiv:1207.4747*, 2012.
- [19] C. L. Lawson and R. J. Hanson. *Solving Least Squares Problems*. Prentice–Hall, Englewood Cliffs, NJ, 1974. Reissued with a survey on recent developments by SIAM, Philadelphia, 1995.
- [20] J. Liu, S. J. Wright, C. Ré, V. Bittorf, and S. Sridhar. An asynchronous parallel stochastic coordinate descent algorithm. *arXiv:1311.1873*, 2013.
- [21] Z.-Q. Luo and P. Tseng. On the convergence of the coordinate descent method for convex differentiable minimization. *Journal of Optimization Theory and Applications*, 72(1):7–35, 1992.
- [22] I. Necoara and A. Patrascu. A random coordinate descent algorithm for optimization problems with composite objective function and linear coupled constraints. *Comp. Opt. and Appl.*, 57(2):307–337, 2014.
- [23] I. Necoara, Y. Nesterov, and F. Glineur. A random coordinate descent method on large optimization problems with linear constraints. Technical report, Technical Report, University Politehnica Bucharest, 2011, 2011.
- [24] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM J. on Optimization*, 19(4):1574–1609, Jan. 2009. ISSN 1052-6234. doi: 10.1137/070704277.
- [25] Y. Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. Core discussion papers, Université catholique de Louvain, Center for Operations Research and Econometrics (CORE), 2010.
- [26] Y. Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.
- [27] J. C. Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. Technical report, ADVANCES IN KERNEL METHODS - SUPPORT VECTOR LEARNING, 1998.
- [28] B. T. Polyak. *Introduction to Optimization*. Optimization Software Inc., 1987. Nov 2010 revision.
- [29] L. A. Rastrigin. *Statisticheskie Metody Poiska Ekstremuma (Statistical Extremum Seeking Methods)*. Nauka, Moscow, 1968.
- [30] B. Recht, C. Re, S. J. Wright, and F. Niu. Hogwild: A Lock-Free Approach to Parallelizing Stochastic Gradient Descent. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. C. N. Pereira, and K. Q. Weinberger, editors, *NIPS*, pages 693–701, 2011.
- [31] P. Richtárik and M. Takáč. Distributed coordinate descent method for learning with big data. *ArXiv e-prints*, Oct. 2013.
- [32] P. Richtárik and M. Takáč. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *arXiv:1107.2848v1*, July 2011.
- [33] P. Richtárik and M. Takáč. Parallel coordinate descent methods for big data optimization. *arXiv:1212.0873v1*, Dec 2012.
- [34] A. Saha and A. Tewari. On the nonasymptotic convergence of cyclic coordinate descent methods. *SIAM Journal on Optimization*, 23(1):576–601, 2013.
- [35] S. Shalev-Shwartz and T. Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. *JMLR*, 14, 2013.
- [36] S. Shalev-Shwartz and T. Zhang. Accelerated mini-batch stochastic dual coordinate ascent. *arXiv:1305.2581v1*, 2013.
- [37] R. Tappenden, P. Richtárik, and J. Gondzio. Inexact coordinate descent: complexity and preconditioning. *arXiv:1304.5530*, 2013.
- [38] R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(1):91–108, 2005.
- [39] P. Tseng and S. Yun. A block-coordinate gradient descent method for linearly constrained nonsmooth separable optimization. *Journal of Optimization Theory and Applications*, 2009.
- [40] P.-W. Wang and C.-J. Lin. Iteration complexity of feasible descent methods for convex optimization. *JMLR*, 15:1523–1548, 2014.

---

# An Upper Bound on the Global Optimum in Parameter Estimation

---

**Khaled S. Refaat** and **Adnan Darwiche**  
Computer Science Department  
University of California, Los Angeles  
{krefaat, darwiche}@cs.ucla.edu

## Abstract

Learning graphical model parameters from incomplete data is a non-convex optimization problem. Iterative algorithms, such as Expectation Maximization (EM), can be used to get a local optimum solution. However, little is known about the quality of the learned local optimum, compared to the unknown global optimum. We exploit variables that are always observed in the dataset to get an upper bound on the global optimum which can give insight into the quality of the parameters learned by estimation algorithms.

## 1 Introduction

Probabilistic graphical models (PGMs) have been useful to many fields, including computer vision, bioinformatics, natural language processing, and statistical physics; see [20, 32, 17, 22]. A graphical model represents a joint probability distribution compactly using a structure populated with parameters. In this paper, we consider two types of graphical models: Markov Random Fields (MRFs) and Bayesian networks (BNs).

An MRF consists of an undirected graph defining conditional independence relationships between variables, and a factor for every maximal clique in the graph; see [15, 16, 24]. A Bayesian network consists of a directed acyclic graph associated with conditional probability tables; see [4].

Learning graphical model parameters from data is typically reduced to finding the maximum likelihood parameters: ones that maximize the probability of a dataset, due to their attractive statistical properties [6]. However, due to the complexity of learning maximum likelihood parameters, other simplified methods have also been proposed in literature such as pseudo-likelihood [2], ratio matching [10], composite maximum likelihood [30], contrastive divergence [9], and more recently the LAP algorithm [23].

A key distinction is commonly drawn between complete and incomplete datasets. In a complete dataset, the value of each variable is known in every example in the dataset, whereas in an incomplete dataset, some variables may have missing values. Computationally, learning from incomplete data can be much harder than learning from complete data, as we discuss next.

When the data is complete, learning maximum likelihood parameters can be done efficiently in BNs by one pass through the dataset, and by solving a convex optimization problem in MRFs. However, in MRFs, evaluating the objective or computing the gradient requires doing inference, to compute the partition function, which is #P-hard [27]. Iterative algorithms, such as gradient descent [28], conjugate gradient (CG) [8], L-BFGS [21], iterative proportional fitting (IPF) [13], and more recently EDML [25] can be used to get the global optimum solution.

On the other hand, if the data is incomplete, the optimization problem is generally non-convex, i.e. has multiple local optima. Iterative algorithms, such as expectation maximization (EM) [5, 18] and gradient descent can be used to get a local optimum solution; see Chapter 19 in [24]. The fixed points of these algorithms correspond to the stationary points of the likelihood function. Hence, these algorithms are not guaranteed to converge to global optima. As such, they are typically applied to multiple seeds (initial parameter estimates), while retaining the best estimates obtained across all seeds. However, little is known about the quality of the learned estimates, compared to the unknown global optimum.

In this paper, we propose an upper bound on the unknown global optimum that can give insight into the quality of the learned estimates, as compared to the global optimum. It may also help derive branch-and-bound methods to get the global optimum. Our proposed technique exploits variables that are always observed and requires solving a convex optimization problem. In case of BNs, this convex optimization problem can be solved efficiently.

The paper is organized as follows. In Section 2, we de-

fine our notation and give an introduction to the problem of learning graphical model parameters. We propose the MRF and BN upper bounds in Sections 3, and 4, respectively. The experimental results are given in Section 5. We review some of the related work in Section 6, and conclude in Section 7.

## 2 Learning Parameters

In this section, we define our notation, and review how parameter estimation for graphical models is formulated as an optimization problem.

### 2.1 Notation

Upper case letters ( $X$ ) denote variables and lower case letters ( $x$ ) denote their values. Variable sets are denoted by bold-face upper case letters ( $\mathbf{X}$ ) and their instantiations by bold-face lower case letters ( $\mathbf{x}$ ).

We use  $\theta$  to denote the set of all network parameters. Parameter learning in graphical models is the process of estimating these parameters  $\theta$  from a given dataset.

A *dataset* is a multi-set of *examples*. Each example is an instantiation of some network variables. We will use  $\mathcal{D}$  to denote a dataset and  $\mathbf{d}_1, \dots, \mathbf{d}_N$  to denote its  $N$  examples. The following is a dataset over four binary variables:

| example | $E$ | $B$       | $A$       | $C$ |
|---------|-----|-----------|-----------|-----|
| 1       | $e$ | $\bar{b}$ | $a$       | ?   |
| 2       | ?   | $\bar{b}$ | $\bar{a}$ | ?   |
| 3       | $e$ | $\bar{b}$ | $\bar{a}$ | ?   |

This dataset has three examples,  $\mathbf{d}_1$ ,  $\mathbf{d}_2$  and  $\mathbf{d}_3$ . For a binary variable  $X$ , we will use  $x$  and  $\bar{x}$  to denote its two values. Moreover, a “?” indicates a missing value of a variable in an example. The first example above corresponds to instantiation  $e, \bar{b}, a$ , while the second example corresponds to instantiation  $\bar{b}, \bar{a}$ .

A variable  $X$  is *fully observed* in a dataset iff the value of  $X$  is known in each example of the dataset (i.e., “?” cannot appear in the column corresponding to variable  $X$ ). Variables  $A$  and  $B$  are fully observed in the above dataset. Moreover, a variable  $X$  is *hidden* in a dataset iff its value is unknown in every example of the dataset (i.e., only “?” appears in the column of variable  $X$ ). Variable  $C$  is hidden in the above dataset. When all variables are fully observed in a dataset, the dataset is said to be *complete*. Otherwise, the dataset is *incomplete*. The above dataset is incomplete. Finally, we will use  $\mathcal{D}_{\mathbf{O}}$  to denote the dataset which results from removing variables outside  $\mathbf{O}$  from dataset  $\mathcal{D}$ .

### 2.2 Markov Random Fields

An MRF is an undirected graph over variables  $\mathbf{X}$ , populated with factors. The MRF parameters are given by the

vector  $\theta = (\dots, \theta_{\mathbf{X}_f}, \dots)$ , where  $\mathbf{X}_f$  are the variables of factor  $f$ . Component  $\theta_{\mathbf{X}_f}$  is a parameter set for a factor  $f$ , assigning a number  $\theta_{\mathbf{X}_f} > 0$  for each instantiation  $\mathbf{x}_f$  of variable set  $\mathbf{X}_f$ .

Given a dataset  $\mathcal{D}$  with examples  $\mathbf{d}_1, \dots, \mathbf{d}_N$ , the *log likelihood* of parameter estimates  $\theta$  is defined as:

$$\ell\ell(\theta|\mathcal{D}) = \sum_{i=1}^N \log Z_{\theta}(\mathbf{d}_i) - N \log Z_{\theta}. \quad (1)$$

Here,  $Z_{\theta}$  is the partition function,  $Z_{\theta} = \sum_{\mathbf{x}} \prod_f \theta_{\mathbf{x}_f}$  and  $Z_{\theta}(\mathbf{d}_i) = \sum_{\mathbf{x} \sim \mathbf{d}_i} \prod_f \theta_{\mathbf{x}_f}$  ( $\mathbf{d}_i \sim \mathbf{x}$  means that instantiations  $\mathbf{d}_i$  and  $\mathbf{x}$  are compatible). For simplicity, we will assume a tabular representation of factors as opposed to an exponential representation as given in [24, Chapter 19]. In our experiments, however, we use the exponential representation to avoid the need for explicit non-negativity constraints.

The first term in Equation 1 is called the data term, whereas the second term is called the model term. If the data is complete, Equation 1 can be formulated as a convex optimization problem, and the data term becomes trivial to evaluate. However, when the data is incomplete, Equation 1 is non-convex.

### 2.3 Bayesian Networks

A Bayesian network is a directed acyclic graph populated with conditional probability tables (CPTs). Generally, we will use  $X$  to denote a variable in a Bayesian network and  $\mathbf{U}$  to denote its parents. For every variable instantiation  $x$  and parent instantiation  $\mathbf{u}$ , the Bayesian network includes a parameter  $\theta_{x|\mathbf{u}}$  that represents the probability  $Pr(X=x|\mathbf{U}=\mathbf{u})$ . This implies the requirement that  $\sum_x \theta_{x|\mathbf{u}} = 1$ , for each parent instantiation  $\mathbf{u}$ .

Given a dataset  $\mathcal{D}$  with examples  $\mathbf{d}_1, \dots, \mathbf{d}_N$ , the *log likelihood* of parameter estimates  $\theta$  is defined as:

$$\ell\ell(\theta|\mathcal{D}) = \sum_{i=1}^N \log Pr_{\theta}(\mathbf{d}_i).$$

Here,  $Pr_{\theta}$  is the distribution induced by the network structure and parameters  $\theta$ . One typically seeks maximum likelihood parameters

$$\theta^* = \underset{\theta}{\operatorname{argmax}} \ell\ell(\theta|\mathcal{D}).$$

It is not uncommon to also assume a Dirichlet prior on network parameters. In particular, for each variable  $X$  with values  $x_1, \dots, x_n$ , and parent instantiation  $\mathbf{u}$ , a Dirichlet prior is specified using exponents  $\psi_{x_1|\mathbf{u}}, \dots, \psi_{x_n|\mathbf{u}}$ . This prior induces a density  $\propto \prod_{i=1}^n [\theta_{x_i|\mathbf{u}}]^{\psi_{x_i|\mathbf{u}} - 1}$  over the parameters  $\theta_{x_1|\mathbf{u}}, \dots, \theta_{x_n|\mathbf{u}}$  of variable  $X$  given parent instantiation  $\mathbf{u}$ . It is also common to assume that exponents are  $> 1$ , which guarantees a unimodal density. With

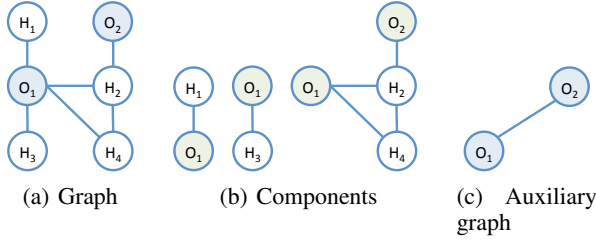


Figure 1: Auxiliary MRF graph under fully observed variables  $\mathbf{O} = \{O_1, O_2\}$ .

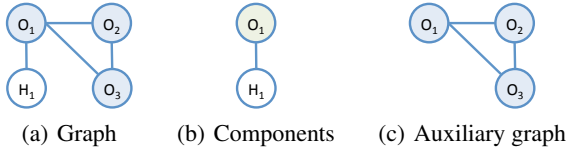


Figure 2: Auxiliary MRF graph under fully observed variables  $\mathbf{O} = \{O_1, O_2, O_3\}$ .

Dirichlet priors, the objective function becomes

$$\ell\ell(\theta|\mathcal{D}) + \log \rho(\theta).$$

Here,  $\rho(\theta)$  is proportional to the prior density on parameters  $\theta$ , and is given by

$$\rho(\theta) = \prod_{X \mathbf{u}} \prod_x [\theta_{x|\mathbf{u}}]^{\psi_{x|\mathbf{u}} - 1}.$$

Parameters that optimize the above objective function are called MAP estimates as they maximize the posterior density of the parameters given the dataset.

When every exponent  $\psi_{x|\mathbf{u}}$  is equal to 1 (uninformative prior), we get  $\rho(\theta) = 1$  and MAP estimates reduce to maximum likelihood estimates. Moreover, when every exponent  $\psi_{x|\mathbf{u}}$  is equal to 2, MAP estimates reduce to maximum likelihood estimates with Laplace smoothing. This is a common technique to deal with the problem of insufficient counts (i.e., instantiations that never appear in the dataset, leading to zero probabilities and division by zero). We will use Laplace smoothing in our experiments.

### 3 An Upper Bound for MRFs

In this section, we utilize variables that are always observed in a dataset to obtain an upper bound on the likelihood. The bound is obtained by solving a convex optimization problem over an auxiliary MRF, which is defined next.

#### 3.1 Decomposition

The auxiliary MRF is obtained by first decomposing the MRF graph  $G$  using variables  $\mathbf{O}$  that are fully observed in the dataset.

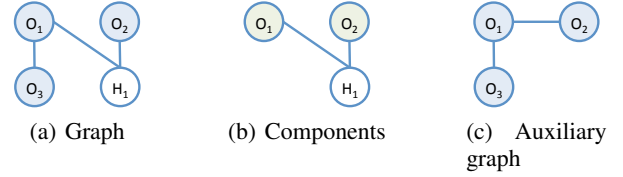


Figure 3: Auxiliary MRF graph under fully observed variables  $\mathbf{O} = \{O_1, O_2, O_3\}$ .

**Definition 1** Let  $G|\mathbf{O}$  be the result of deleting variables  $\mathbf{O}$  from graph  $G$ . A component of  $G|\mathbf{O}$  is a maximal set of nodes  $\mathbf{S}$  that are connected in  $G|\mathbf{O}$ . A variable  $B$  is a boundary for component  $\mathbf{S}$  iff edge  $B - S$  appears in  $G$ ,  $B \notin \mathbf{S}$  and  $S \in \mathbf{S}$ .

Boundary variables must be included in  $\mathbf{O}$ . Moreover, component variables cannot intersect with  $\mathbf{O}$ . Figures 1–3 depict the components and boundaries of some MRF graphs.

We are now ready to define the auxiliary MRF by defining its graph. The auxiliary MRF will then have one factor over each maximal clique of this graph.

**Definition 2** The auxiliary graph for graph  $G$  and variables  $\mathbf{O}$  is denoted  $A_{G|\mathbf{O}}$  and defined as follows: (1) The nodes of  $A_{G|\mathbf{O}}$  are the variables  $\mathbf{O}$ ; (2)  $A_{G|\mathbf{O}}$  has an edge  $X - Y$  iff the edge exists in  $G$  or  $X$  and  $Y$  are boundary variables of some component of  $G|\mathbf{O}$ .

Figures 1–3(c) depict some auxiliary MRF graphs.

#### 3.2 Optimization

We will next use the auxiliary MRF graph to formulate a convex optimization problem, called the auxiliary problem. The solution of this auxiliary problem will provide an upper bound on the likelihood.

**Definition 3** Given an MRF graph  $G$ , and a corresponding dataset  $\mathcal{D}$  with fully observed variables  $\mathbf{O}$ , the auxiliary optimization problem is that of learning the parameters of auxiliary MRF  $A_{G|\mathbf{O}}$  from dataset  $\mathcal{D}_{\mathbf{O}}$ .

The auxiliary optimization problem is always convex. This follows since the graph  $A_{G|\mathbf{O}}$  contains only variables  $\mathbf{O}$ , which are fully observed in the dataset  $\mathcal{D}$ . Hence, the auxiliary optimization problem corresponds to learning the parameters of an MRF under complete data ( $\mathcal{D}_{\mathbf{O}}$  is a complete dataset in this case).

The following theorem shows that the solution of the convex optimization problem provides an upper bound on the likelihood.

**Theorem 1** Let  $G$  be an MRF graph and  $\mathcal{D}$  be a corre-



sponding dataset with fully observed variables  $\mathbf{O}$ . Let  $f(\theta)$  be the likelihood function for MRF graph  $G$ , and let  $g(\theta)$  be the likelihood function for its auxiliary MRF graph  $A_{G|\mathbf{O}}$ . We then have  $f(\theta) \leq g(\theta^*)$ , where  $\theta^*$  is the global optimum for  $g(\theta)$ .

**Proof** Let  $F_1(\mathbf{X}_1), \dots, F_n(\mathbf{X}_n)$  be the factors of auxiliary MRF  $A_{G|\mathbf{O}}$ , representing parameters  $\theta$  (i.e., each  $F_j(\mathbf{x}_j)$  is a parameter in  $\theta$ ). Note that  $\mathbf{X}_j$  is a maximal clique of the auxiliary graph and  $\mathbf{X}_j \subseteq \mathbf{O}$ . Let the dataset  $\mathcal{D}$  be  $\{\mathbf{d}_1, \dots, \mathbf{d}_N\}$ . The convex optimization problem  $g(\theta)$  is then

$$\text{maximize } g(\theta) = \sum_{i=1}^N \log Z_{\theta}(\mathbf{d}_i) - N \log Z_{\theta} \quad (2)$$

where

$$Z_{\theta} = \sum_{\mathbf{o}} \prod_{j=1}^n F_j(\mathbf{x}_j), \quad \mathbf{x}_j \sim \mathbf{o} \quad (3)$$

$$Z_{\theta}(\mathbf{d}_i) = \prod_{j=1}^n F_j(\mathbf{x}_j), \quad \mathbf{x}_j \sim \mathbf{d}_i \quad (4)$$

We will now expand the above equations for optimizing the auxiliary likelihood  $g(\theta)$  so we can optimize the original likelihood  $f(\theta)$ . The basic observation is that  $f(\theta)$  can be written in terms of marginals over the fully observed variables. However, these marginals are not free to take any values, as they have to correspond to some original parameters that realize such marginals. Hence, we must constrain these marginals, which correspond to auxiliary parameters  $F_i(\mathbf{x}_i)$ , in terms of the original parameters. We do this next.

First, note that for each factor  $f_k(\mathbf{Y}_k)$  of the original MRF  $G$ , there is some factor  $F_j(\mathbf{X}_j)$  of the auxiliary MRF, such that  $\mathbf{Y}_k \cap \mathbf{O} \subseteq \mathbf{X}_j$ . We will therefore assign each original factor  $f_k$  to a corresponding auxiliary factor  $F_j$ , writing  $f_k^j$  to denote this assignment.

Next, for each auxiliary factor  $F_j(\mathbf{X}_j)$ , let  $\mathbf{Z}_j$  be the variables appearing in original factors  $f_k^j$ , but not in the auxiliary factor  $F_j$ . Consider now the following equation, which defines auxiliary parameters  $F_j(\mathbf{x}_j)$  in terms of original parameters  $f_k^j(\mathbf{y}_k)$ :

$$F_j(\mathbf{x}_j) = \sum_{\mathbf{z}_j} \prod_k f_k^j(\mathbf{y}_k), \quad \mathbf{y}_k \sim \mathbf{x}_j \mathbf{z}_j \quad (5)$$

The original optimization problem  $f(\theta)$  can now be defined using Equations 2, 3 and 4, subject to the equality constraints of Equation 5 (i.e., we are now optimizing the original parameters  $f_k^j(\mathbf{y}_k)$ ). By relaxing these equality constraints, and optimizing over the auxiliary parameters  $F_j(\mathbf{x}_j)$ , we get back the auxiliary optimization problem. Since the latter is obtained by relaxing constraints, we have  $f(\theta) \leq g(\theta^*)$ .  $\square$



Figure 4: A chain MRF with alternating fully observed variables, and its corresponding auxiliary MRF.

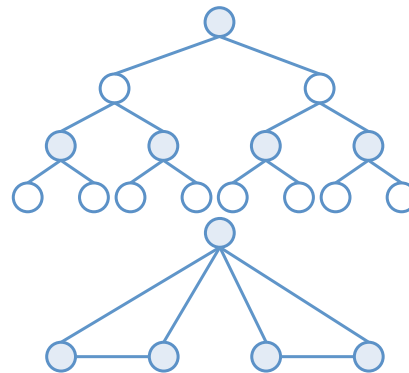


Figure 5: A binary tree MRF with alternating fully observed levels, and its corresponding auxiliary MRF.

Note that when all variables are fully observed in the dataset  $\mathcal{D}$  (i.e., the dataset is complete), the auxiliary MRF graph corresponds to the original MRF graph, and the bound becomes exact.

### 3.3 Computing the Bound

The proposed upper bound can be computed using standard methods for estimating parameters under complete data. These methods require inference on the auxiliary MRF, whose complexity depends on the treewidth of its underlying graph. This treewidth can be larger or smaller than the treewidth of the original MRF, depending on the patterns of data incompleteness. We will illustrate this next using a set of examples, in which fully observed nodes are shaded, while hidden nodes are left unshaded.

Figures 4 and 5 show MRFs of bounded treewidth and certain patterns of data incompleteness that lead to auxiliary MRFs with bounded treewidth. In particular, Figure 4 shows a chain with alternating fully observed and hidden variables, which results in an auxiliary MRF with treewidth 1, regardless of the chain length. Figure 5 shows a complete binary tree with alternating fully observed levels, leading to an auxiliary MRF with treewidth 2, regardless of the tree depth.

Figure 6 shows an example where the auxiliary MRF has a lower treewidth. However, Figure 7 shows an  $n \times n$  grid, leading to an auxiliary MRF with treewidth  $2n - 1$ . Similarly, Figure 8 shows an MRF with treewidth 1, yet an auxiliary MRF of treewidth  $n$ .

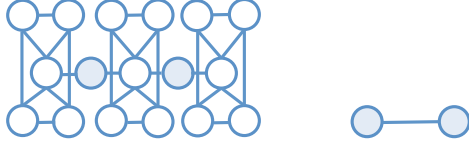


Figure 6: An MRF structure that leads to an auxiliary MRF of lower treewidth.

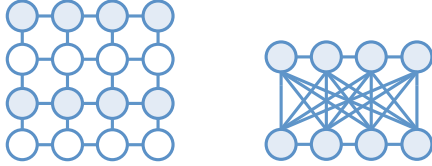


Figure 7: A grid with alternating fully observed rows, and its corresponding auxiliary MRF.

#### 4 An Upper Bound for Bayesian Networks

We now present a similar upper bound on the likelihood function for a Bayesian network structure  $G$ . Again, the bound is defined based on the set of fully observed variables  $\mathbf{O}$  in a dataset.

**Definition 4 ([26])** Let  $G|\mathbf{O}$  be the result of deleting edges in DAG  $G$  that are outgoing from variables  $\mathbf{O}$ . A component of  $G|\mathbf{O}$  is a maximal set of variables  $\mathbf{S}$  that are connected in  $G|\mathbf{O}$ . A variable  $B$  is a boundary for component  $\mathbf{S}$  iff edge  $B \rightarrow S$  appears in  $G$ ,  $B \notin \mathbf{S}$  and  $S \in \mathbf{S}$ .

Figure 9 depicts an example DAG with its components and boundaries. Note that the boundary variables  $\mathbf{B}$  of a component must all be fully observed,  $\mathbf{B} \subseteq \mathbf{O}$ . Moreover, for any component  $\mathbf{S}$ , the variables  $\mathbf{S} \cap \mathbf{O}$  must be leaf nodes in  $G|\mathbf{O}$ .

We will next interpret the boundary variables  $\mathbf{B}$  of each component  $\mathbf{S}$  as the parents of observed variables in component  $\mathbf{S}$ . This interpretation will be used to define an auxiliary distribution over the fully observed variables.

**Definition 5** The auxiliary distribution for DAG  $G$  and variables  $\mathbf{O}$  is denoted  $P_{G|\mathbf{O}}$  and defined as follows: (1)  $P_{G|\mathbf{O}}$  is over the variables  $\mathbf{O}$ , (2)  $P_{G|\mathbf{O}}$  is the product of factors  $Pr(\mathbf{L}|\mathbf{B})$ , where  $\mathbf{B}$  is the boundary variables of some component  $\mathbf{S}$  and  $\mathbf{L} = \mathbf{S} \cap \mathbf{O}$ .



Figure 8: An MRF structure with treewidth 1, and its corresponding auxiliary MRF of treewidth  $n$ .

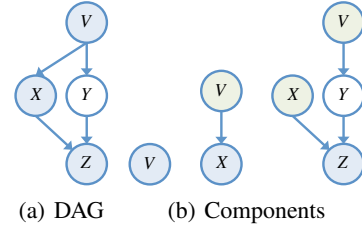


Figure 9: A DAG and its components under fully observed variables  $\mathbf{O} = \{V, X, Z\}$ .

Each probability  $Pr(\mathbf{l}|\mathbf{b})$  will be interpreted as an auxiliary parameter  $\theta_{\mathbf{l}|\mathbf{b}}$ . We can now state our upper bound.

**Theorem 2** Let  $G$  be a DAG and  $\mathcal{D}$  be a corresponding dataset with fully observed variables  $\mathbf{O}$ . Let  $f(\theta)$  be the likelihood function for  $G$  and  $\mathcal{D}$ , and let  $g(\theta)$  be the likelihood function for the auxiliary distribution  $P_{G|\mathbf{O}}$ . We then have  $f(\theta) \leq g(\theta^*)$ , where  $\theta^*$  is the global optimum for  $g(\theta)$ .

**Proof** We now sketch the proof of this theorem, which is similar to the one for MRFs. That is, we express auxiliary parameters in terms of original parameters, allowing us to formulate the original optimization problem as an optimization problem with non-convex equality constraints (which relate auxiliary and original parameters). By relaxing these equality constraints, we obtain a convex optimization problem that provides an upper bound on the original optimization problem. Hence, it suffices to show the non-convex equality constraints in this case.

Consider a factor  $Pr(\mathbf{X}|\mathbf{U})$  of the auxiliary distribution, and the corresponding parameters  $\theta_{\mathbf{x}|\mathbf{u}}$ . Variables  $\mathbf{X}$  must then be leaves of some component  $\mathbf{S}$  in  $G|\mathbf{O}$ , and  $\mathbf{U}$  must correspond to the boundary variables of component  $\mathbf{S}$ . One can then express each auxiliary parameter  $\theta_{\mathbf{x}|\mathbf{u}}$  in terms of original parameters that pertain only to the variables in component  $\mathbf{S}$ . In particular, let  $Pr(\cdot)$  be the distribution induced by the original DAG  $G$  and let  $\mathbf{y}$  be an instantiation of variables  $\mathbf{S} \setminus \mathbf{X}$ . We then have  $Pr(\mathbf{x}|\mathbf{u}) = \sum_{\mathbf{y}} Pr(\mathbf{x}, \mathbf{y}|\mathbf{u})$ . Moreover,  $Pr(\mathbf{x}, \mathbf{y}|\mathbf{u})$  can be expressed in terms of original parameters pertaining only to variables  $\mathbf{S}$ . This follows since, given  $\mathbf{U}$ ,  $\mathbf{S}$  is independent of all other variables in DAG  $G$ .  $\square$

One difference from the upper bound for MRFs is that this bound can be computed more efficiently. In particular, the optimal estimate  $\theta^*$  can be identified using a single pass through the dataset  $\mathcal{D}_{\mathbf{O}}$ . Similarly,  $g(\theta^*)$  can be computed using a single pass through the dataset, once the estimate  $\theta^*$  is identified.

## 5 Experimental Results

Our experiments are structured as follows. Given a network  $G$ , we generate a dataset  $\mathcal{D}$  while ensuring that a certain percentage of variables are fully observed, with all others hidden. Using dataset  $\mathcal{D}$ , we estimate the parameters of network  $G$  using EM.

We compare the local optimum learned by EM, to the proposed bound gotten using decomposition, and to the bound that assumes all distributions are valid (which we call the naive bound).

The naive bound is computed by discarding the graph structure and assigning a probability to every data example based on its number of occurrences in the dataset. This effectively assumes a fully connected graph. Consider, for example, a simple dataset with a data example  $\mathbf{d}_1$  that is repeated twice and another  $\mathbf{d}_2$  that is repeated 3 times. The naive bound assigns a probability  $\frac{2}{5}$  to  $\mathbf{d}_1$ , and  $\frac{3}{5}$  to  $\mathbf{d}_2$ , and computes the likelihood:  $(\frac{2}{5})^2 \times (\frac{3}{5})^3$ .

Before we present our results, we have the following observations on our data generation model. First, we made all unobserved variables hidden (as opposed to missing at random) as this leads to a more difficult learning problem, especially for EM. Second, it is not uncommon to have a significant number of variables that are always observed in real-world datasets. For example, in the UCI repository: the internet advertisements dataset has 1558 variables, only 3 of which have missing values; the automobile dataset has 26 variables, where 7 have missing values; the dermatology dataset has 34 variables, where only age can be missing; and the mushroom dataset has 22 variables, where only one variable has missing values [1].

In our experiments, we use the following networks: alarm, andes, asia, win95pts, diagnose, pigs, spect, water, together with chains, trees, and grids. Network win95pts (76 variables) is an expert system for printer troubleshooting in Windows 95, whereas Network pigs is used for diagnosing the PSE disease. Network andes is an intelligent tutoring system. Network diagnose is from the UAI 2008 evaluation. Network spect is a naive Bayes network induced from a dataset in the UCI ML repository, with 1 class variable and 22 attributes. Chains, trees, and grids are randomly generated networks. The other networks are commonly used benchmarks.

Figures from 10 to 25 show the objective function values gotten by EM for different benchmarks and for different percentages of fully observed variables, together with the proposed bound (decomposition bound), and the naive bound. We have the following observations on the results.

In most cases, our proposed bound was much tighter than the naive bound. One can also see that the proposed bound coincides (or almost coincides) with the EM’s curve in Fig-

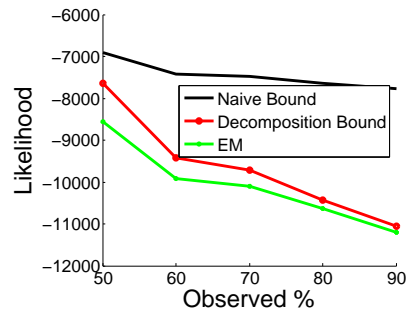


Figure 10: Upper bound for network Alarm.

ures 11, 15, 18, 19, 20, 21, 22, 23, 24, and 25. This shows that the bound can be tight in many cases. When the bound coincides with the EM curve, it provides a certificate that EM is getting the global optimum in these cases.

Moreover, in Figures 10, 14, and 17, as the number of fully observed variables increases, the gap between the proposed bound and EM’s curve tends to shrink, which suggests that the bound becomes tight and that EM gets close to the global optimum in these cases. On the other hand, for cases where the proposed bound was not close to the EM’s curve, it could be that EM is getting a local optimum, or the bound is not tight, in these cases.

Furthermore, we note that the excellent performance of the upper bound on Network Spect in Figure 15, and on tree networks in Figures 20, 21, and 22 is partially because hidden variables associated with leaf nodes in these networks can be ignored from the computation of the likelihood, as their values are summed out.

We finally conduct an experiment to see how often EM approaches the bound if started from different seeds; using a  $3 \times 3$  grid while fully observing 50% of the variables. Figure 26 shows the difference in likelihood between the upper bound and EM for this benchmark, when started from different seeds (x-axis). One can see that EM gets close to the bound in many cases for this benchmark. We note, however, that a more comprehensive study is needed for assessing the quality of EM estimates under different seeds—a study that can be significantly aided by the proposed upper bound.

## 6 Related Work

Decomposing Bayesian networks based on fully observed variables was proposed in [26] to speed-up parameter estimation. Our bound relies on this decomposition as a first step in formulating the auxiliary optimization problem.

Variational methods (see [14]) can provide lower bounds on the likelihood in graphical models. Moreover, an upper bound for the likelihood in the context of Gaussian mixtures was proposed in [3]. However, this bound only works

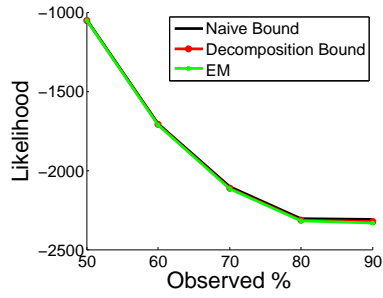


Figure 11: Upper bound for network Asia.

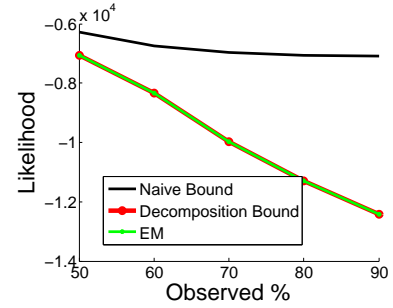


Figure 15: Upper bound for network Spect.

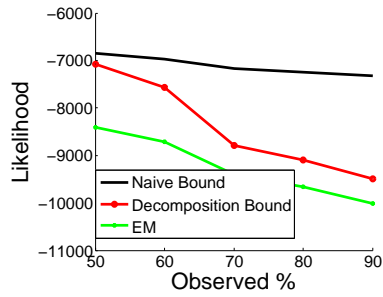


Figure 12: Upper bound for network Win95pts.

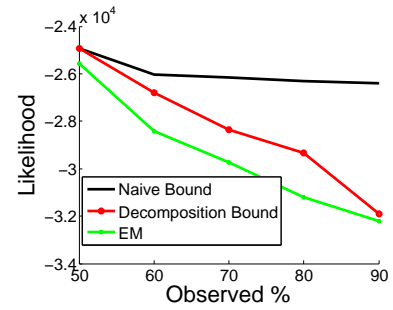


Figure 16: Upper bound for network Water.

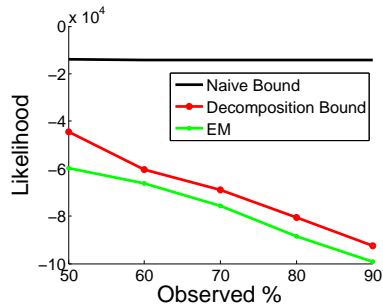


Figure 13: Upper bound for network Diagnose.

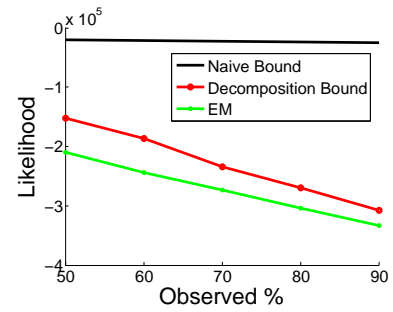


Figure 17: Upper bound for network Pigs.

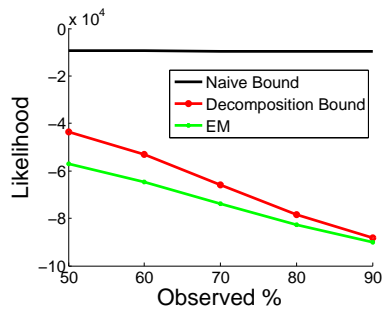


Figure 14: Upper bound for network Andes.

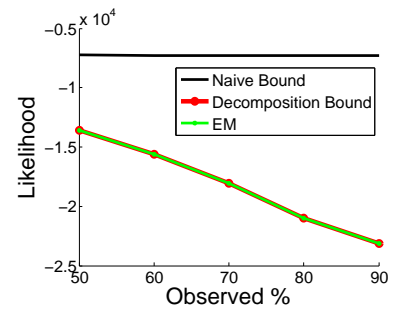


Figure 18: Upper bound for a chain network (50 nodes).

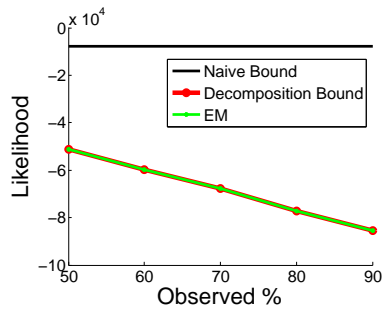


Figure 19: Upper bound for a chain network (180 nodes).

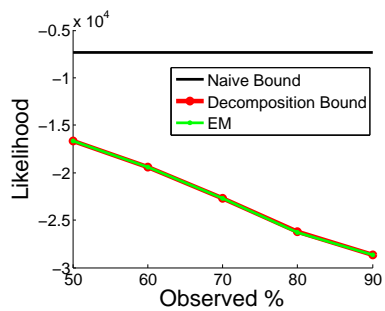


Figure 20: Upper bound for a tree network (63 nodes).

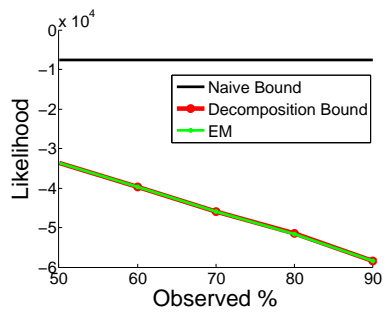


Figure 21: Upper bound for a tree network (127 nodes).

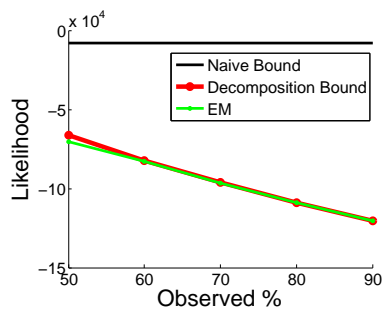


Figure 22: Upper bound for a tree network (255 nodes).

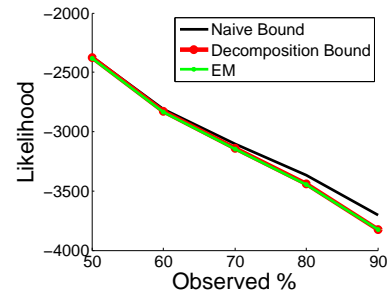


Figure 23: Upper bound for a  $3 \times 3$  MRF grid.

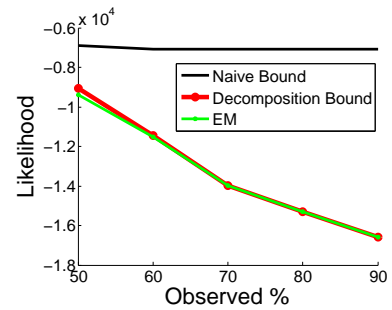


Figure 24: Upper bound for a  $6 \times 6$  MRF grid.

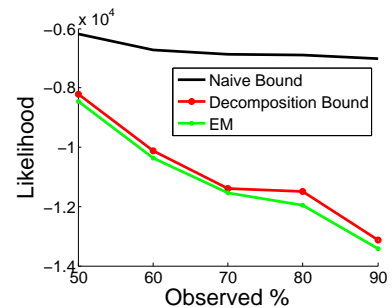


Figure 25: Upper bound for a  $9 \times 9$  MRF grid.

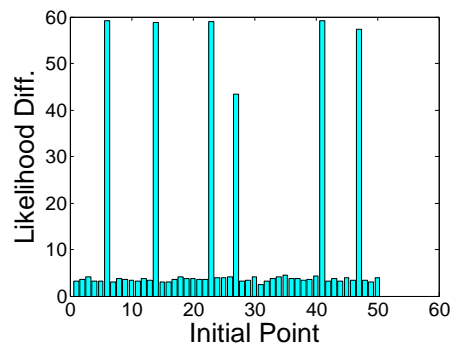


Figure 26: Likelihood difference between decomposition bound and EM started from different points.

asymptotically. An upper bound on maximum likelihood that only works for phylogenetic trees was proposed in [7]. Techniques for computing upper and lower bounds on likelihoods in sigmoid and noisy-OR networks were proposed in [12].

Some work also exists for obtaining upper and lower bounds on the partition function. In particular, mean field theory, e.g. [33, 14], provides such a lower bound (tighter bounds have also been derived [19]). In contrast, upper bounds are not widely available [31]. For the special case of the Ising Model, a recursive procedure was proposed for upper bounding the log partition function [11]. An upper bound on the partition function of an arbitrary MRF was proposed in [31] based on solving a convex variational problem. While bounds on the partition function can be used to get an upper bound on the likelihood, the non-convex term related to the data remains non-convex, which does not make the bound easy to compute. The bound we proposed, however, is based on solving a convex optimization problem.

## 7 Conclusion

We proposed a technique for obtaining an upper bound on the global optimum in parameter estimation. The technique applies to incomplete datasets and exploits variables that are always observed in the dataset. The bound is computed by solving a convex optimization problem, which can be solved by a single pass through the dataset in Bayesian networks. The proposed bound can be useful in providing a certificate of global optimality for parameters learned by estimation algorithms. Empirically, we showed that the bound can be tight, and can be used to show that an estimation algorithm is obtaining the global optimum or an estimate that is very close to the optimum.

## Acknowledgments

This work was supported by ONR grant #N00014-12-1-0423 and NSF grant #IIS-1118122.

## References

- [1] K. Bache and M. Lichman. Uci machine learning repository. Technical report, Irvine, CA: University of California, School of Information and Computer Science, 2013.
- [2] J. Besag. Statistical Analysis of Non-Lattice Data. *The Statistician*, 24:179–195, 1975.
- [3] Christophe Biernacki. An asymptotic upper bound of the likelihood to prevent gaussian mixtures from degenerating. Technical report, Université de Franche-Comté, 2004.
- [4] Adnan Darwiche. *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press, 2009.
- [5] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39:1–38, 1977.
- [6] R. A. Fisher. On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society of London Series*, 1922.
- [7] Michael D. Hendy and Barbara R. Holland. Upper bounds on maximum likelihood for phylogenetic trees. *Bioinformatics*, 2003.
- [8] Magnus R. Hestenes and Eduard Stiefel. Methods of conjugate gradients for solving linear systems. *Research of the National Bureau of Standards*, 1952.
- [9] G. Hinton. Training products of experts by minimizing contrastive divergence. In *Neural Computation*, 2000.
- [10] A. Hyvärinen. Estimation of non-normalized statistical models using score matching. *JMLR*, 2005.
- [11] T. S. Jaakkola and M. Jordan. Recursive algorithms for approximating probabilities in graphical models. In *Advances in Neural Information Processing Systems*, 1996.
- [12] Tommi S. Jaakkola and Michael I. Jordan. Computing upper and lower bounds on likelihoods in intractable networks. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 1998.
- [13] Radim Jirousek and Stanislav Preucil. On the effective implementation of the iterative proportional fitting procedure. *Computational Statistics & Data Analysis*, 19(2):177–189, 1995.
- [14] M. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. Saul. *Learning in Graphical Models*, chapter “An introduction to variational methods for graphical models. Cambridge, MA: MIT Press, 1999.
- [15] R. Kindermann and J. L. Snell. *Markov Random Fields and their Applications*. American Mathematical Society, 1980.
- [16] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.
- [17] J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001.
- [18] S. L. Lauritzen. The EM algorithm for graphical association models with missing data. *Computational Statistics and Data Analysis*, 19:191–201, 1995.
- [19] S. L. Lauritzen. *Graphical Models*. Oxford, U.K.: Oxford Univ. Press, 1996.
- [20] S Z. Li. Markov random field modeling in image analysis. *Springer-Verlag*, 2001.
- [21] D. C. Liu and J. Nocedal. On the Limited Memory BFGS Method for Large Scale Optimization. *Mathematical Programming*, 45(3):503–528, 1989.
- [22] E. Marinari, G. Parisi, and J.J. Ruiz-Lorenzo. Numerical simulations of spin glass systems. *Spin Glasses and Random Fields*, 1997.

- [23] Yariv Dror Mizrahi, Misha Denil, and Nando de Freitas. Linear and parallel learning of Markov random fields. In *In International Conference on Machine Learning (ICML)*, 2014.
- [24] Kevin Patrick Murphy. *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.
- [25] Khaled S. Refaat, Arthur Choi, and Adnan Darwiche. EDML for learning parameters in directed and undirected graphical models. In *Advances in Neural Information Processing Systems 26*, pages 1502–1510, 2013.
- [26] Khaled S. Refaat, Arthur Choi, and Adnan Darwiche. Decomposing parameter estimation problems. In *Advances in Neural Information Processing Systems 27*, pages 1565–1573, 2014.
- [27] Dan Roth. On the hardness of approximate reasoning. *Artificial Intelligence*, 1996.
- [28] S. Russel, J. Binder, D. Koller, and K. Kanazawa. Local learning in probabilistic networks with hidden variables. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, 1995.
- [29] R. Shachter. Evidence absorption and propagation through evidence reversals. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 1990.
- [30] C. Varin, N. Reid, and D Firth. An overview of composite likelihood methods. *Statistica Sinica*, 2011.
- [31] Martin J. Wainwright, Tommi S. Jaakkola, and IEEE Alan S. Willsky, Fellow. A new class of upper bounds on the log partition function. *IEEE Transactions on Information Theory*, 2005.
- [32] C. Yanover, O. Schueler-Furman, and Y. Weiss. Minimizing and learning energy functions for side-chain prediction. In *Speed, Terry and Huang, Haiyan (eds.), Research in Computational Molecular Biology, volume 4453 of Lecture Notes in Computer Science*, 2007.
- [33] J. Zhang. The application of the gibbs-bogoliubov-feynman inequality in mean-field calculations for Markov random-fields. *IEEE Tran. on Image Process.*, 5(7):1208–1214, 1996.

---

# A Markov Game Model for Valuing Player Actions in Ice Hockey

---

**Kurt Routley**  
School of Computing Science  
Simon Fraser University  
Vancouver, BC, Canada  
kdr4@sfu.ca

**Oliver Schulte**  
School of Computing Science  
Simon Fraser University  
Vancouver, BC, Canada  
oschulte@cs.sfu.ca

## Abstract

A variety of advanced statistics are used to evaluate player actions in the National Hockey League, but they fail to account for the context in which an action occurs or to look ahead to the long-term effects of an action. We apply the Markov Game formalism to develop a novel approach to valuing player actions in ice hockey that incorporates context and lookahead. Dynamic programming is used to learn Q-functions that quantify the impact of actions on goal scoring resp. penalties. Learning is based on a massive dataset that contains over 2.8M events in the National Hockey League. The impact of player actions is found to vary widely depending on the context, with possible positive and negative effects for the same action. We show that lookahead makes a substantial difference to the action impact scores. Players are ranked according to the aggregate impact of their actions. We compare this impact ranking with previous player metrics, such as plus-minus, total points, and salary.

## 1 INTRODUCTION

A fundamental goal of sports statistics is to understand which actions contribute to winning in what situation. As sports have entered the world of big data, there is increasing opportunity for large-scale machine learning to model complex sports dynamics. The research described in this paper applies AI techniques to model the dynamics of ice hockey; specifically the Markov Game model formalism [Littman, 1994], and related computational techniques such as the dynamic programming value iteration algorithm. We make use of a massive dataset about matches in the National Hockey League (NHL). This dataset comprises all play-by-play events from 2007 to 2014, for a total of over 2.8M events/actions and almost 600K play sequences. The

Markov Game model comprises over 1.3M states. Whereas most previous works on Markov Game models aim to compute optimal strategies or policies [Littman, 1994] (i.e., minimax or equilibrium strategies), we learn a model of how hockey is actually played, and do not aim to compute optimal strategies. In reinforcement learning (RL) terminology, we use dynamic programming to compute an *action-value* Q-function in the *on policy* setting [Sutton and Barto, 1998]. In RL notation, the expression  $Q(s, a)$  denotes the expected reward of taking action  $a$  in state  $s$ .

**Motivation** Motivation for learning a Q-function for NHL hockey dynamics includes the following.

*Knowledge Discovery.* The Markov Game model provides information about the likely consequences of actions. The basic model and algorithms can easily be adapted to study different outcomes of interest, such as goals and penalties.

*Player Evaluation.* One of the main tasks for sports statistics is evaluating the performance of players [Schumaker et al., 2010]. A common approach is to assign action values, and sum the corresponding values each time a player takes the respective action. An advantage of this additive approach is that it provides highly interpretable player rankings. A simple and widely used example in ice hockey is the +/- score: for each goal scored by (against) a player's team when he is on the ice, add +1 (-1) point. Researchers have developed several extensions of +/- for hockey [Macdonald, 2011; Spagnola, 2013; Schuckers and Curro, 2013].

There are two major problems with the previous action count approaches used in ice hockey. (1) They are unaware of the *context* of actions within a game. For example, a goal is more valuable in a tied-game situation than when the scorer's team is already four goals ahead [Pettigrew, 2015]. Another example is that if a team manages two successive shots on goal, the second attempt typically has a higher chance of success. In the Markov Game model,  $context = state$ . Formally, the Q function depends *both* on the state  $s$  and the action  $a$ . Richer state spaces therefore capture more of the context of an action. (2) Previous ac-



tion scores are based on immediate positive consequences of an action (e.g. goals following a shot). However, an action may have medium-term and/or ripple effects rather than immediate consequences in terms of visible rewards like goals. Therefore evaluating the impact of an action requires *lookahead*. Long-term lookahead is especially important in ice hockey because evident rewards like goals occur infrequently [Lock and Schuckers, 2009]. For example, if a player receives a penalty, this leads to a manpower disadvantage for his team, known as a powerplay for the other team. It is easier to score a goal during a powerplay, but this does not mean that a goal will be scored immediately after the penalty. For another example, if a team loses the puck in their offensive zone, the resulting counterattack by the other team may lead to a goal eventually but not immediately. The dynamic programming value iteration algorithm of Markov Decision Processes provides a computationally efficient way to perform unbounded lookahead.

**Evaluation** Our evaluation learns Q-functions for two reward events, scoring the next goal and receiving the next penalty. We observe a wide variance of the impact of actions with respect to states, showing context makes a substantial difference. We provide examples of the context dependence to give a qualitative sense of how the Markov Game model accounts for context. To evaluate player performance, we use the Q-function to quantify the value of a player’s action in a context. The action values are then aggregated over games and seasons to get player impact scores. Player impact scores correlate with plausible alternative scores, such as a player’s total points, but improve on these measures, as our impact score is based on many more events.

**Contributions** We make our extensive dataset available on-line, in addition to our code and the learned Markov game model [Routley et al., 2015]. The main contributions of this paper may be summarized as follows:

1. The first Markov Game model for a large ice hockey state space (over 1.3M), based on play sequence data.
2. Learning a Q-function that models play dynamics in the National Hockey League from a massive data set (2.8M events). We introduce a variant of AD-Trees as a data structure to (1) compute and store the large number of sufficient statistics required [Moore and Lee, 1998], and (2) support value iteration updates.
3. Applying the Q-function to define a context-aware look-ahead measure of the value of an action, over configurable objective functions (rewards).
4. Applying the context-aware action values to score hockey player actions, including how players affect penalties as well as goals.

**Paper Organization.** We review related work in measuring player contributions and machine learning in sports in Section 2. We then give some background information on the ice hockey domain and NHL play-by-play sequences data. Our Markov Game model translates the hockey domain features into the Markov formalism. We describe how we implement a scalable value iteration for the ice hockey domain. The evaluation section addresses the impact of context and lookahead. We apply the model to rank the aggregate performance of players and describe the resulting player ranking. We view our work as taking the first step, not the last, in applying AI modelling techniques to ice hockey. Therefore, we conclude with a number of potential extensions and open problems for future work.

## 2 RELATED WORK

**Evaluating Actions and Players in Ice Hockey** Several papers aim to improve the basic +/- score with statistical techniques [Macdonald, 2011; Gramacy et al., 2013; Spagnola, 2013]. A common approach is to use regression techniques where an indicator variable for each player is used as a regressor for a goal-related quantity (e.g., log-odds of a goal for the player’s team vs. the opposing team). The regression weight measures the extent to which the presence of a player contributes to goals for his team or prevents goals for the other team. These approaches look at only goals, no other actions. The only context they take into account is which players are on the ice when a goal is scored. Regression could be combined with our Markov game model to capture how team impact scores depend on the presence or absence of individual players.

The closest predecessor to our work in ice hockey is the Total Hockey Rating (THoR) [Schuckers and Curro, 2013]. This assigns a value to all actions, not only goals. Actions were evaluated based on whether or not a goal occurred in the following 20 seconds after an action. This work used data from the 2006/2007 NHL season only. THoR assumes a fixed value for every action and does not account for the context in which an action takes place. Furthermore, the window of 20 seconds restricts the lookahead value of each action. Our Q-learning method is not restricted to any particular time window for lookahead.

**Expected Possession Value** [Cervone et al., 2014] uses spatial-temporal tracking data for basketball to build the POINTWISE model for valuing player decisions and player actions. Conceptually, their approach to defining action values is the closest predecessor to ours: The counterpart to the value of a state in a Markov game is called expected possession value (EPV). The counterpart to the impact of an action on this value is called EPV-added (EPVA). Cervone *et al.* emphasize the broad potential of the context-based impact definitions: “we assert that most questions that coaches, players, and fans have about basketball, par-

ticularly those that involve the offense, can be phrased and answered in terms of EPV.”

While the definition of action impact is conceptually very similar, Cervone et al. use neither AI terminology nor AI techniques, which we cover in this paper. Moreover, all the underlying details are different between our model and theirs: The NHL does not yet have and therefore we do not use spatial tracking data, which is the main focus of Cervone et al.. Cervone et al. discuss the advantages of using a discrete state space for stochastic consistency, but consider it computationally infeasible for their data. We show that leveraging AI data structures and algorithms makes handling a large discrete state space feasible for ice hockey. Including the local action history in the state space allows us to capture the medium-term effects of actions. This is more important for ice hockey than for basketball, because scoring in basketball occurs at much shorter intervals.

**Markov Decision Process Models for Other Sports** MDP-type models have been applied in a number of sports settings, such as baseball, soccer and football. For review, please see Cervone et al. [2014]. Our work is similar in that our method uses value iteration on a Markovian state space, however, previous Markov models in sports use a much smaller state space. The goal of these models is to find an optimal policy for a critical situation in a sport or game. In contrast, we learn in the on-policy setting whose aim is to model hockey dynamics as it is actually played.

### 3 DOMAIN DESCRIPTION: HOCKEY RULES AND HOCKEY DATA

We outline the rules of hockey and describe the dataset available from the NHL.

#### 3.1 HOCKEY RULES

We give a brief overview of rules of play in the NHL [National Hockey League, 2014]. NHL games consist of three periods, each 20 minutes in duration. A team has to score more goals than their opponent within three periods in order to win the game. If the game is still tied after three periods, the teams will enter a fourth overtime period, where the first team to score a goal wins the game. If the game is still tied after overtime during the regular season, a shootout will commence. During the playoffs, overtime periods are repeated until a team scores a goal to win the game. Teams have five skaters and one goalie on the ice during even strength situations. Penalties result in a player sitting in the penalty box for two, four, or five minutes and the penalized team will be shorthanded, creating a manpower differential between the two teams. The period where one team is penalized is called a powerplay for the opposing team with a manpower advantage. A shorthanded

goal is a goal scored by the penalized team, and a powerplay goal is a goal scored by the team on the powerplay.

#### 3.2 DATA FORMAT

The NHL provides information about sequences of play-by-play events, which are scraped from <http://www.nhl.com> and stored in a relational database. The real-world dataset is formed from 2,827,467 play-by-play events recorded by the NHL for the complete 2007-2014 seasons, regular season and playoff games, and the first 512 games of the 2014-2015 regular season. A breakdown of this dataset is shown in Table 1. The type of events recorded by the NHL from the 2007-2008 regular season and onwards are listed in Table 2. There are two types of events: actions performed by players and start and end markers for each play sequence. Every event is marked with a continuous timestamp, and every action is also marked with a zone  $Z$  and which team, Home or Away, carries out the action.

Table 1: Size of Dataset

|                            |           |
|----------------------------|-----------|
| <b>Number of Teams</b>     | 32        |
| <b>Number of Players</b>   | 1,951     |
| <b>Number of Games</b>     | 9,220     |
| <b>Number of Sequences</b> | 590,924   |
| <b>Number of Events</b>    | 2,827,467 |

Table 2: NHL Play-By-Play Events Recorded

| <b>Action Event</b> | <b>Start/End Event</b>   |
|---------------------|--------------------------|
| Faceoff             | Period Start             |
| Shot                | Period End               |
| Missed Shot         | Early Intermission Start |
| Blocked Shot        | Penalty                  |
| Takeaway            | Stoppage                 |
| Giveaway            | Shootout Completed       |
| Hit                 | Game End                 |
| Goal                | Game Off                 |
|                     | Early Intermission End   |

### 4 MARKOV GAMES

A Markov Game [Littman, 1994], sometimes called a stochastic game, is defined by a set of states,  $S$ , and a collection of action sets, one for each agent in the environment. State transitions are controlled by the current state and one action from each agent. For each agent, there is an associated reward function mapping a state transition to a reward. An overview of how our Markov Game model fills in this schema is as follows. There are two players, the Home Team  $H$  and the Away Team  $A$ . In each state,

only one team performs an action, although not in a turn-based sequence. This reflects the way the NHL records actions. Thus at each state of the Markov Game, exactly one player chooses No-operation. State transitions follow a semi-episodic model [Sutton and Barto, 1998] where play moves from episode to episode, and information from past episodes is recorded as a list of *context features*. The past information includes the goal score and manpower. A sequence in the NHL play-by-play data corresponds to an episode in Markov decision process terminology. *Within* each episode/sequence, our game model corresponds to a game tree with perfect information as used in AI game research [Russell and Norvig, 2010]. We introduce the following generic notation for all states, following [Russell and Norvig, 2010; Littman, 1994].

- $Occ(s)$  is the number of occurrences of state  $s$  as observed in the play-by-play data.
- $Occ(s, s')$  is the number of occurrences of state  $s$  being immediately followed by state  $s'$  as observed in the play-by-play data.  $(s, s')$  forms an edge in the transition graph of the Markov Game model.
- The transition probability function  $TP$  is a mapping of  $S \times S \rightarrow (0, 1]$ . We estimate it using the observed transition frequency  $\frac{Occ(s, s')}{Occ(s)}$ .

We begin by defining context features, then play sequences.

#### 4.1 STATE SPACE: CONTEXT FEATURES

Previous work on Markov process models for ice hockey [Thomas et al., 2013] defined states in terms of hand-selected features that are intuitively relevant for the game dynamics, such as the goal differential and penalties. We refer to such features as **context features**. Context features remain the same throughout each play sequence.

Table 3: Context Features

| Notation | Name                  | Range     |
|----------|-----------------------|-----------|
| $GD$     | Goal Differential     | $[-8, 8]$ |
| $MD$     | Manpower Differential | $[-3, 3]$ |
| $P$      | Period                | $[1, 7]$  |

A **context state** lists the values of relevant features at a point in the game. These features are shown in Table 3, together with the range of integer values observed. Goal Differential  $GD$  is calculated as Number of Home Goals - Number of Away Goals. A positive (negative) goal differential means the home team is leading (trailing). Manpower Differential  $MD$  is calculated as Number of Home Skaters on Ice - Number of Away Skaters on Ice. A positive manpower differential typically means the home team

is on the powerplay (away team is penalized), and a negative manpower differential typically means the home team is shorthanded (away team is on the powerplay).<sup>1</sup> Period  $P$  represents the current period number the play sequence occurs in, typically ranging in value from 1 to 5. Periods 1 to 3 are the regular play of an ice hockey game, and periods 4 and onwards are for overtime and shootout periods as needed.

Potentially, there are  $(17 \times 7 \times 7) = 833$  context states. In our NHL dataset, 450 context states occur at least once. Table 4 includes statistics for the top-20 context states over all 590,924 play sequences, and lists 52,793 total goals and 89,612 total penalties. Positive differences are for the home team and negative differences are for the away team. For example, a Goal Difference of 7.1% means the home team is 7.1% more likely to score a goal in that context state than the away team. Similarly, a Penalty Difference of -33.2% means the away team is 33.2% more likely to receive a penalty in that context state than the home team. Our Markov model is very well calibrated, due to the frequency estimation method, meaning that its predictions match the observed frequencies of goals and penalties. We explain below how the model predictions are computed.

A number of previous papers on hockey dynamics have considered the context features of play sequences. The important trends that it is possible to glean from statistics such as those shown in Table 4 have been discussed in several papers. Our data analysis confirms these observations on a larger dataset than previously used. Notable findings include the following.

1. Home team advantage: the same advantages in terms of context features translate into higher scoring rates.
2. Penalties are more frequent than goals, except for the overtime period 4 (cf. [Schuckers and Brozowski, 2012]).
3. Gaining a powerplay substantially increases the conditional probability of scoring a goal [Thomas et al., 2013].
4. Gaining a powerplay also significantly increases the conditional probability of receiving a penalty [Schuckers and Brozowski, 2012].
5. Shorthanded goals are surprisingly likely: a manpower advantage translates only into a goal scoring difference of at most 64.8%, meaning the shorthanded team scores the next goal with a conditional probability of 17.6%. (Home team powerplay when  $P = 1$ .)

While such patterns provide interesting and useful insights into hockey dynamics, they do not consider action events.

<sup>1</sup>Pulling the goalie can also result in a skater manpower advantage.

Table 4: Statistics for Top-20 Most Frequent Context States. GD = Goal Differential, MD = Manpower Differential, P = Period.

| GD | MD | P | #Sequences | #Goals | #Penalties | Observed        |                    | Model Predicts  |                    |
|----|----|---|------------|--------|------------|-----------------|--------------------|-----------------|--------------------|
|    |    |   |            |        |            | Goal Difference | Penalty Difference | Goal Difference | Penalty Difference |
| 0  | 0  | 1 | 78,118     | 5,524  | 11,398     | 7.06%           | -2.26%             | 7.06%           | -2.26%             |
| 0  | 0  | 2 | 38,315     | 2,935  | 5,968      | 7.60%           | -2.92%             | 7.60%           | -2.92%             |
| 0  | 0  | 3 | 30,142     | 2,050  | 3,149      | 5.85%           | -2.19%             | 5.85%           | -2.19%             |
| 1  | 0  | 2 | 29,662     | 2,329  | 4,749      | 2.02%           | 2.17%              | 2.02%           | 2.17%              |
| 1  | 0  | 3 | 25,780     | 2,076  | 3,025      | 4.34%           | 3.54%              | 4.34%           | 3.54%              |
| -1 | 0  | 2 | 25,498     | 1,970  | 4,044      | 8.63%           | -8.70%             | 8.63%           | -8.70%             |
| 1  | 0  | 1 | 24,721     | 1,656  | 4,061      | 5.31%           | 3.42%              | 5.31%           | 3.42%              |
| -1 | 0  | 3 | 22,535     | 1,751  | 2,565      | 0.74%           | -18.28%            | 0.74%           | -18.28%            |
| -1 | 0  | 1 | 20,813     | 1,444  | 3,352      | 4.57%           | -8.05%             | 4.57%           | -8.05%             |
| 2  | 0  | 3 | 17,551     | 1,459  | 2,286      | 6.92%           | -0.87%             | 6.92%           | -0.87%             |
| 2  | 0  | 2 | 15,419     | 1,217  | 2,620      | 2.71%           | 2.90%              | 2.71%           | 2.90%              |
| -2 | 0  | 3 | 13,834     | 1,077  | 1,686      | -2.32%          | -12.57%            | -2.32%          | -12.57%            |
| 0  | 1  | 1 | 12,435     | 1,442  | 2,006      | 64.77%          | 31.70%             | 64.77%          | 31.70%             |
| -2 | 0  | 2 | 11,799     | 882    | 1,927      | 3.85%           | -15.72%            | 3.85%           | -15.72%            |
| 0  | -1 | 1 | 11,717     | 1,260  | 2,177      | -54.76%         | -44.79%            | -54.76%         | -44.79%            |
| 3  | 0  | 3 | 10,819     | 678    | 1,859      | 0.29%           | 1.24%              | 0.29%           | 1.24%              |
| -3 | 0  | 3 | 7,569      | 469    | 1,184      | 7.04%           | -6.25%             | 7.04%           | -6.25%             |
| 0  | 1  | 2 | 7,480      | 851    | 1,157      | 56.99%          | 25.67%             | 56.99%          | 25.67%             |
| 0  | 0  | 4 | 7,024      | 721    | 535        | 5.69%           | -10.65%            | 5.69%           | -10.65%            |
| 0  | -1 | 2 | 6,853      | 791    | 1,150      | -52.47%         | -37.39%            | -52.47%         | -37.39%            |

This means that analysis at the sequence level does not consider the internal dynamics within each sequence, and that it is not suitable for evaluating the impact of hockey actions. We next extend our state space to include actions.

#### 4.2 STATE SPACE: PLAY SEQUENCES

We expand our state space with actions and action histories. The basic set of 8 possible actions is listed in Table 2. Each of these actions has two parameters: which team  $T$  performs the action and the zone  $Z$  where the action takes place. Zone  $Z$  represents the area of the ice rink in which an action takes place.  $Z$  can have values Offensive, Neutral, or Defensive, relative to the team performing an action. For example,  $Z = \text{Offensive zone}$  relative to the home team is equivalent to  $Z = \text{Defensive zone}$  relative to the away team. A specification of an action plus parameters is an **action event**. Using action description language notation [Levesque et al., 1998], we write action events in the form  $a(T, Z)$ . For example,  $\text{faceoff}(\text{Home}, \text{Neutral})$  denotes the home team wins a faceoff in the neutral zone. We usually omit the action parameters from generic notation and write  $a$  for a generic action event.

A **play sequence**  $h$  is a sequence of events starting with exactly one start marker, followed by a list of action events, and ended by at most one end marker. Start and end markers are shown in Table 2, adding shots and faceoffs as start markers, and goals as end markers. We also allow empty history  $\emptyset$  as a valid play sequence. A **complete** play sequence ends with an end marker. A **state** is a pair

$s = \langle \mathbf{x}, h \rangle$  where  $\mathbf{x}$  denotes a list of context features and  $h$  an action history. State  $s$  represents a play sequence consisting of action events  $a_1, a_2, \dots, a_n$  and with a particular  $GD, MD,$  and  $P$  as the context. If the sequence  $h$  is empty, then state  $s$  is purely a context node. Table 5 shows an example of a NHL play-by-play action sequence in tabular form. Potentially, there are  $(7 \times 2 \times 3)^{40} = 42^{40}$  action histories. In our dataset, 1,325,809 states, that is, combinations of context features and action histories, occur at least once. We store sequence data in SQL tables (see Table 5). SQL provides fast retrieval, and native support for the necessary COUNT operations.

Table 5: Sample Play-By-Play Data in Tabular Format

| GameId | Period | Sequence Number | Event Number | Event                       |
|--------|--------|-----------------|--------------|-----------------------------|
| 1      | 1      | 1               | 1            | PERIOD START                |
| 1      | 1      | 1               | 2            | faceoff(Home,Neutral)       |
| 1      | 1      | 1               | 3            | hit(Away,Neutral)           |
| 1      | 1      | 1               | 4            | takeaway(Home,Defensive)    |
| 1      | 1      | 1               | 5            | missed_shot(Away,Offensive) |
| 1      | 1      | 1               | 6            | shot(Away,Offensive)        |
| 1      | 1      | 1               | 7            | giveaway(Away,Defensive)    |
| 1      | 1      | 1               | 8            | takeaway(Home,Offensive)    |
| 1      | 1      | 1               | 9            | missed_shot(Away,Offensive) |
| 1      | 1      | 1               | 10           | goal(Home,Offensive)        |
| 1      | 1      | 2               | 11           | faceoff(Away,Neutral)       |
| ...    |        |                 |              |                             |

#### 4.3 STATE TRANSITIONS

If  $h$  is an incomplete play sequence, we write  $h \star a$  for the play sequence that results from appending  $a$  to  $h$ , where  $a$  is an action event or an end marker. Similarly if  $s = \langle \mathbf{x}, h \rangle$ ,

then  $s \star a \equiv \langle \mathbf{x}, h \star a \rangle$  denotes the unique successor state that results from executing action  $a$  in  $s$ . This notation utilizes the fact that context features do not change until an end marker is reached. For example, the goal differential does not change unless a goal event occurs. If  $h$  is a complete play sequence, then the state  $\langle \mathbf{x}, h \rangle$  has a unique successor  $\langle \mathbf{x}', \emptyset \rangle$ , where the mapping from  $\mathbf{x}$  to  $\mathbf{x}'$  is determined by the end marker. For instance, if the end marker is  $goal(Home, *)$ , then the goal differential increases by 1. A sample of our state transition graph is shown in Figure 1. Note that  $R(s)$  is the reward value for the state, and will be discussed in Section 4.4. In Figure 1, the reward encodes the objective of scoring a goal.

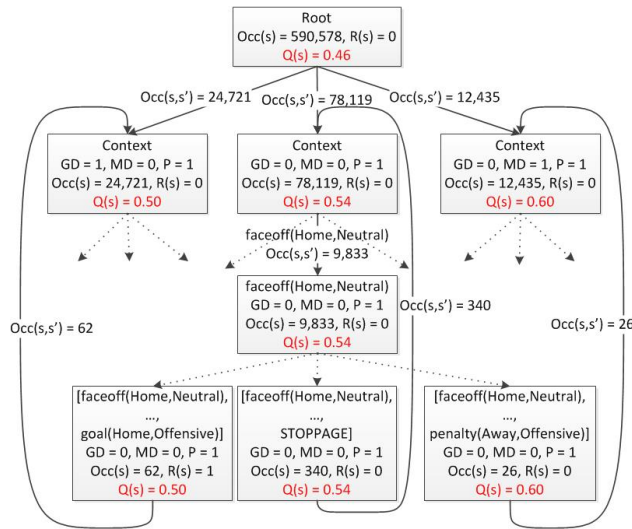


Figure 1: State Transition Graph

Since the complete action history is encoded in the state, action-state pairs are equivalent to state pairs. For example, we can write  $Q(s \star a)$  to denote the expected reward from taking action  $a$  in state  $s$ , where  $Q$  maps states to real numbers, rather than mapping action-state pairs to real numbers, as is more usual.

#### 4.4 REWARD FUNCTIONS: NEXT GOAL AND NEXT PENALTY

A strength of Markov Game modelling is value iteration can be applied to many reward functions depending on what results are of interest. We focus on two: scoring the next goal, and receiving the next penalty (a cost rather than a reward). These are two important events that change the course of an ice hockey game. For example, penalties affect goal scoring differentials, as shown in Table 4. Penalties are also one path to goals that a coach may want to understand in more detail. For instance, if a team receives an unusual number of penalties, a coach may want to know which players are responsible and by which actions. The next goal objective can be represented in the Markov Game

model as follows.

1. For any state  $s$  with a complete play sequence that ends in a Home resp. Away goal, we set  $R_H(s) := 1$  resp.  $R_A(s) := 1$ . For other states the reward is 0.
2. Any state  $s$  with a complete play sequence that ends in a Home resp. Away goal is an absorbing state (no transitions from this state).

With these definitions, the expected reward represents the probability that if play starts in state  $s$ , a random walk through the state space of unbounded length ends with a goal for the Home team resp. the Away team. The cost function for Receiving the Next Penalty can be represented in exactly the same way.

## 5 CONSTRUCTING THE STATE TRANSITION GRAPH

The main computational challenge is to build a data structure for managing the state space. The state space is large because each (sub)sequence of actions defines a new state. Since we are modelling the actual hockey dynamics in the on policy setting, we need consider only action sequences observed in some NHL match, rather than the much larger space of all possible action sequences. We use the classic AD-tree structure [Moore and Lee, 1998] to compute and store sufficient statistics over observed action sequences. The AD-tree is a tree of play sequences where a node is expanded only with those successors observed in at least one match. The play sequence tree is augmented with additional edges that model further state transitions; for example, a new action sequence is started after a goal. The augmented AD-tree structure compactly manages sufficient statistics, in this case state transition probabilities. It also supports value iteration updates very efficiently.

We outline an algorithm for Context-Aware State Transition Graph construction. The root node initializes the graph, and is an empty node with no context or event information. For each node, the context information  $GD$ ,  $MD$ , and  $P$  are set when the new node is created, and the new action  $a$  is added to the sequence along with the zone  $Z$  that  $a$  occurs in. The reward  $R(s)$  is also applied to each node. The node counts  $Occ(s)$  and edge counts  $Occ(s, s')$  are applied to each node and edge respectively, and are used to generate transition probabilities  $TP$  for the value iteration using observed frequencies. The NHL play-by-play event data records goals, but no separate event for the shot leading to the goal exists. Following [Schuckers and Curro, 2013], we record the shot leading to the goal in addition to the goal itself by injecting a shot event into the event sequence prior to the goal.

## 6 VALUE ITERATION

Recall that since states encode action histories, in our model learning the expected value of states is equivalent to learning a Q-function (Section 4.3). In reinforcement learning terms, there is no difference between the value function  $V$  and the Q-function in our model. We can therefore apply standard value iteration over states [Sutton and Barto, 1998] to learn a Q-function for our ice hockey Markov Game. Algorithm 1 shows pseudo-code. We compute separate Q-functions for the Home team and for the Away team. Since we are in the on policy setting, we have a fixed policy for the other team. This means we can treat the other team as part of the environment, and reduce the Markov Game to two single-agent Markov decision processes. In our experiments, we use a relative convergence of 0.0001 as our convergence criterion, and 100,000 as the maximum number of iterations.

---

### Algorithm 1 Dynamic Programming for Value Iteration

---

**Require:** Markov Game model, convergence criterion  $c$ , maximum number of iterations  $M$

```

1:  $lastValue = 0$ 
2:  $currentValue = 0$ 
3:  $converged = false$ 
4: for  $i = 1; i \leq M; i \leftarrow i + 1$  do
5:   for all states  $s$  in the Markov Game model do
6:     if  $converged == false$  then
7:        $Q_{i+1}(s) =$ 
            $R(s) + \frac{1}{Occ(s)} \sum_{(s,s') \in E} (Occ(s,s') \times Q_i(s'))$ 
8:        $currentValue = currentValue + |Q_{i+1}(s)|$ 
9:     end if
10:  end for
11:  if  $converged == false$  then
12:    if  $\frac{currentValue - lastValue}{currentValue} < c$  then
13:       $converged = true$ 
14:    end if
15:  end if
16:   $lastValue = currentValue$ 
17:   $currentValue = 0$ 
18: end for

```

---

## 7 EVALUATION AND RESULTS

We discuss the results of action values in Section 7.1 and player values in Section 7.2.

### 7.1 ACTION IMPACT VALUES

The main quantity we consider is the **impact** of an action as a function of context (= Markov state). This is defined as follows:

$$impact(s, a) \equiv Q_T(s \star a) - Q_T(s)$$

where  $T$  is the team executing the action  $a$ . In a zero-sum game, the state value is usually defined as the final result following optimal play [Russell and Norvig, 2010]. Intuitively, the value specifies which player has a better position in a state. Since we are not modelling optimal play, but actual play in an on policy setting, the expected difference in rewards is the natural counterpart. The impact quantity measures how performing an action in a state affects the expected reward difference. Figure 2 shows a boxplot for the action impact values as they range over different contexts, i.e., states in the Markov Game model. (Boxplots produced with MATLAB R2014a.) The red dots are outliers beyond 2.7 s.d. A cutoff of -0.2 and 0.2, shown by the horizontal dashed line, was used for the impact values on both boxplots. While the Q-values are based on the frequency of states, we weight all states equally in discussing the properties of the Q-function. The boxplot does not include Q-values for states whose frequency is below 5%. It is clear from Figure 2 that *depending on the context and event history, the value of an action can vary greatly*. The context-dependence is observed for both scoring goals and receiving penalties.

**Impact on Scoring the Next Goal.** All actions, with the exception of faceoffs won in the offensive zone, have at least one state where the action has a positive impact, and another state with a negative impact. Two examples of how the value of the same action can depend on the context include the following, which we found by examining states with extreme impact values.

*Blocked Shot.* Blocking the first shot on net when killing a penalty decreases a team’s scoring rate ( $impact = -0.0864$ ). But blocking the second shot on net increases the scoring rate ( $impact = 0.1399$ ).

*Penalty.* Receiving a penalty when on the powerplay is very bad ( $impact = -0.1789$ ). But if a player, while on the penalty kill, receives a penalty while goading their opponent into an offsetting penalty, the penalty actually increases their team’s scoring rate ( $impact = 0.0474$ ).

The THoR player ratings compute the impact of actions based on goals that immediately follow the action ([Lock and Schuckers, 2009; Schuckers et al., 2011]; see Section 2). The values given for each action in [Lock and Schuckers, 2009] are displayed as an asterisk in Figure 2(a). The THoR values agree with our median impact values in terms of whether an action generally has positive or negative impact. For example, penalties are known to generally be good for the opposing team, and shots are good for the shooter’s team. THoR values are close to the median Markov model values in 6 out of 10 cases. This comparison suggests THoR aggregates action values over many contexts the Markov game models explicitly. In a le-

sion study described in the supplementary material, we examine directly defining the value of an action as the average impact of the action over all states. Using the average impact as a fixed action value leads to a loss of information, as measured by the entropy of the prediction for which team scores the next goal. Another lesion study described in the supplementary material assesses the importance of propagating information between states, especially from one play sequence to subsequent ones. Our results show that goal impact values of the actions change substantially depending on how much information the model propagates.

**Impact on Receiving Penalties.** The range of action values with the probability of the next penalty as the objective function is shown in Figure 2(b). Faceoffs in the Offensive Zone and takeaways cause penalties for the opponent. Giveaways and goals tend to be followed by a penalty for the player’s team. The latter finding is consistent with the observation that there are more penalties called against teams with higher leads [Schuckers and Brozowski, 2012]. A possible explanation is referees are reluctant to penalize a trailing team.

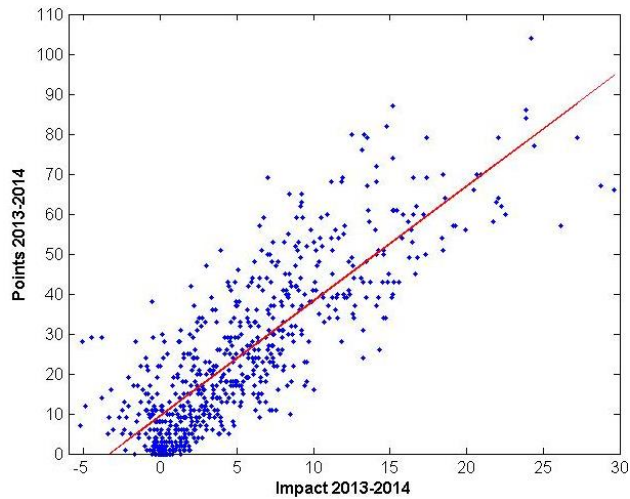


Figure 3: 2013-2014 Player Goal Impact Vs. Season Points

## 7.2 PLAYER VALUATIONS

As players perform actions on behalf of their team, it is intuitive to apply the impact scores of team actions to the players performing the action, yielding player valuations. To calculate player valuations, we apply the impact of an action to the player as they perform the action. Next, we sum the impact scores of a player’s actions over a single game, and then over a single season, to compute a net season impact score for the player. This procedure is equivalent to comparing the actions taken by a specific player to those of the league-average player, similar to previous work [Pettigrew, 2015; Cervone et al., 2014]. We compare

Table 6: 2013-2014 Top-8 Player Impact Scores For Goals

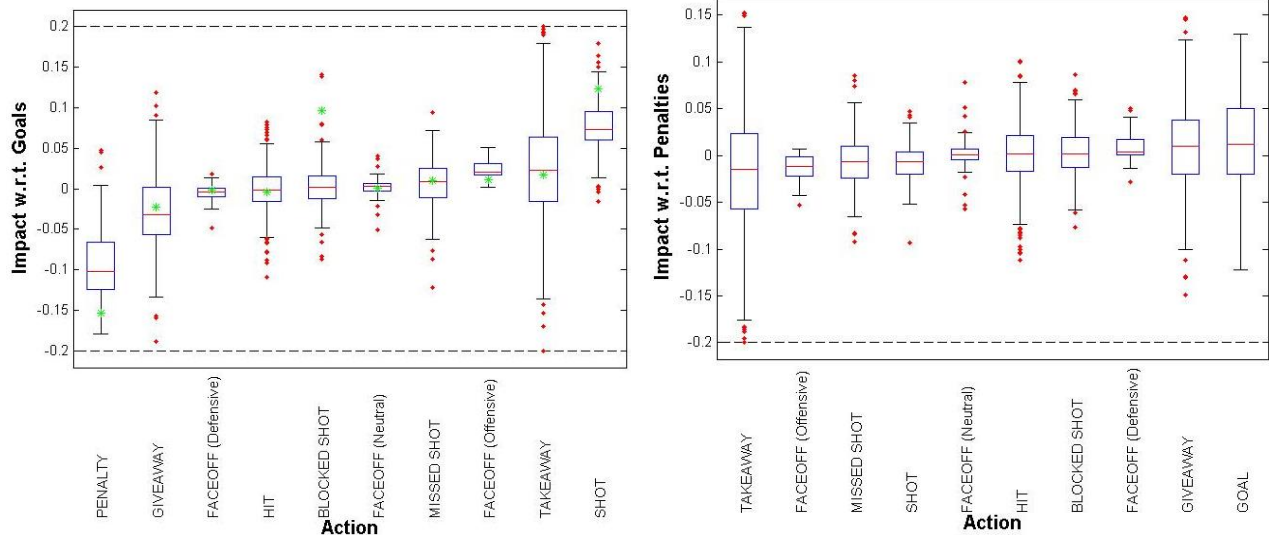
| Name           | Goal Impact | Points | +/- | Salary       |
|----------------|-------------|--------|-----|--------------|
| Jason Spezza   | 29.64       | 66     | -26 | \$5,000,000  |
| Jonathan Toews | 28.75       | 67     | 25  | \$6,500,000  |
| Joe Pavelski   | 27.20       | 79     | 23  | \$4,000,000  |
| Marian Hossa   | 26.12       | 57     | 26  | \$7,900,000  |
| Patrick Sharp  | 24.43       | 77     | 12  | \$6,500,000  |
| Sidney Crosby  | 24.23       | 104    | 18  | \$12,000,000 |
| Claude Giroux  | 23.89       | 86     | 7   | \$5,000,000  |
| Tyler Seguin   | 23.89       | 84     | 16  | \$4,500,000  |

impact on Next Goal Scored with three other player ranking metrics: points earned, salary, and +/- . To avoid confounding effects between different seasons, we use only the most recent full season, 2013-2014. Player impact scores are shown in Table 6. Tables for all seasons are available as well [Routley, 2015]. Figure 3 shows that next goal impact correlates well with points earned. A point is earned for each goal or assist by a player. Since these players have a high impact on goals, they also tend to have a positive +/- rating. Jason Spezza is an anomaly, as he has the highest impact score but a very negative +/- score. This is because his Ottawa team performed poorly overall in the 2013-2014 season: The team overall had a goal differential of -29, one of the highest goal differentials that season. This example shows that impact scores distinguish a player who generally performs useful actions but happens to be on a poor team.

In Table 7, we see player impact with respect to Next Penalty Received. High impact numbers indicate a tendency to cause penalties for a player’s own team, or prevent penalties for the opponent. We compare the Q-function impact numbers to Penalties in Minutes (PIM), +/- , and salary. Players with high Q-function numbers have high penalty minutes as we would expect. They also have low +/- , which shows the importance of penalties for scoring chances. Their salaries tend to be lower. There are however notable exceptions, such as Dion Phaneuf, who draws a high salary although his actions have a strong tendency to incur penalties.

Table 7: 2013-2014 Top-8 Player Impacts For Penalties

| Name            | Penalty Impact | PIM | +/- | Salary      |
|-----------------|----------------|-----|-----|-------------|
| Chris Neil      | 62.58          | 211 | -10 | \$2,100,000 |
| Antoine Roussel | 54.26          | 209 | -1  | \$625,000   |
| Dion Phaneuf    | 52.52          | 144 | 2   | \$5,500,000 |
| Zac Rinaldo     | 48.65          | 153 | -13 | \$750,000   |
| Rich Clune      | 47.08          | 166 | -7  | \$525,000   |
| Tom Sestito     | 46.34          | 213 | -14 | \$650,000   |
| Zack Smith      | 44.55          | 111 | -9  | \$1,500,000 |
| David Perron    | 42.49          | 90  | -16 | \$3,500,000 |



(a) Impact on the probability of Scoring the Next Goal. Higher numbers are *better* for the team that performs the action. (b) Impact on the probability of Receiving the Next Penalty. Higher numbers are *worse* for the team that performs the action.

Figure 2: Action Impact Values vary with context. The central mark is the median, the edges of the box are the 25th and 75th percentiles. The whiskers are at the default value, approximately 2.7 s.d.

## 8 CONCLUSION

We have constructed a Markov Game Model for a massive set of NHL play-by-play events with a rich state space. Tree-based data structures support efficient parameter estimation and storage. Value iteration computes the values of each action given its context and sequence history—the Q-function of the model. Compared to previous work that assigns a single value to actions, the Q-function incorporates two powerful sources of information for valuing hockey actions: (1) It takes into account the context of the action, represented by the Markov Game state. (2) It models the medium-term impact of an action by propagating its effect to future states. Propagating action effects across sequences utilizes the ordering of play sequences in a game, rather than treating sequences as an unordered independent set. Analysis of the computed Q-function shows the impact of an action varies greatly with context, and medium-term ripple effects make a difference. We apply our model to evaluate the performance of players in terms of their actions’ total impact. Impact scores for the next goal correlate with points. The impact of players on the next penalty has to our knowledge not been previously considered, and shows some surprises, as some highly-paid players hurt their team by causing penalties. In sum, the Q-function is a powerful AI concept that captures much information about hockey dynamics as the game is played in the NHL.

**Future Work** The NHL data provides a rich dataset for real-world event modelling. A number of further AI techniques can be applied to utilize even more of the available information than our Markov Game model does. A promis-

ing direction is to extend our discrete Markov Game model with data about continuous quantities. These include (i) the time between events, (ii) the absolute game time of the events, (iii) location of shots [Krzywicki, 2005]. Our use of reinforcement learning techniques has been mainly for finding patterns in a rich data set, in the spirit of descriptive statistics and data mining. Another goal is to *predict* a player or team’s future performance based on past performance using machine learning techniques. For example, sequence modelling would be able to generalize from play sequence information. A promising model class are Piecewise Constant Conditional Intensity Models for continuous time event sequences [Gunawardana et al., 2011; Parikh et al., 2012]. These models are especially well suited for sequences with a large set of possible events, such as our action events. Another extension is to evaluate players with respect to similar players [Cervone et al., 2014], for instance, players who play the same position. A potential future application for improving play and advising coaches is in finding strengths and weaknesses of teams: We can use the Q-function to find situations in which a team’s mix of actions provides a substantially different expected result from that of a generic team.

## Acknowledgements

This work was supported by a Discovery Grant from the National Sciences and Engineering Council of Canada. We received helpful comments from Tim Swartz and anonymous UAI referees. We are grateful for constructive discussions in SFU’s Sports Analytics Research Group. Zeyu Zhang assisted with the preparation of the on-line datasets.



## References

- Cervone, D., D'Amour, A., Bornn, L., and Goldsberry, K. (2014). Pointwise: Predicting points and valuing decisions in real time with nba optical tracking data. In *8th Annual MIT Sloan Sports Analytics Conference, February*, volume 28.
- Gramacy, R., Jensen, S., and Taddy, M. (2013). Estimating player contribution in hockey with regularized logistic regression. *Journal of Quantitative Analysis in Sports*, 9:97–111.
- Gunawardana, A., Meek, C., and Xu, P. (2011). A model for temporal dependencies in event streams. In *Advances in Neural Information Processing Systems*, pages 1962–1970.
- Krzywicki, K. (2005). Shot quality model: A logistic regression approach to assessing nhl shots on goal.
- Levesque, H., Pirri, F., and Reiter, R. (1998). Foundations for the situation calculus. *Linköping Electronic Articles in Computer and Information Science*, 3(18).
- Littman, M. L. (1994). Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the eleventh international conference on machine learning*, volume 157, pages 157–163.
- Lock, D. and Schuckers, M. (2009). Beyond +/-: A rating system to compare nhl players. Presentation at joint statistical meetings.
- Macdonald, B. (2011). An improved adjusted plus-minus statistic for nhl players.
- Moore, A. W. and Lee, M. S. (1998). Cached sufficient statistics for efficient machine learning with large datasets. *J. Artif. Intell. Res. (JAIR)*, 8:67–91.
- National Hockey League (2014). National hockey league official rules 2014-2015.
- Parikh, A. P., Gunawardana, A., and Meek, C. (2012). Conjoint modeling of temporal dependencies in event streams. In *UAI Bayesian Modelling Applications Workshop*.
- Pettigrew, S. (2015). Assessing the offensive productivity of nhl players using in-game win probabilities. In *9th Annual MIT Sloan Sports Analytics Conference*.
- Routley, K. (2015). A markov game model for valuing player actions in ice hockey. Master's thesis, Simon Fraser University.
- Routley, K., Schulte, O., and Zhao, Z. (2015). Q-learning for the nhl. <http://www.cs.sfu.ca/~oschulte/sports/>.
- Russell, S. and Norvig, P. (2010). *Artificial Intelligence: A Modern Approach*. Prentice Hall.
- Schuckers, M. and Brozowski, L. (2012). Referee analytics: An analysis of penalty rates by national hockey league officials. In *MIT Sloan Sports Analytics Conference*.
- Schuckers, M. and Curro, J. (2013). Total hockey rating (thor): A comprehensive statistical rating of national hockey league forwards and defensemen based upon all on-ice events. In *7th Annual MIT Sloan Sports Analytics Conference*.
- Schuckers, M. E., Lock, D. F., Wells, C., Knickerbocker, C. J., and Lock, R. H. (2011). National hockey league skater ratings based upon all on-ice events: An adjusted minus/plus probability (ampp) approach. Unpublished manuscript.
- Schumaker, R. P., Solieman, O. K., and Chen, H. (2010). Research in sports statistics. In *Sports Data Mining*, volume 26 of *Integrated Series in Information Systems*, pages 29–44. Springer US.
- Spagnola, N. (2013). The complete plus-minus: A case study of the columbus blue jackets. Master's thesis, University of South Carolina.
- Sutton, R. S. and Barto, A. G. (1998). *Reinforcement learning: an introduction*. MIT Press, Cambridge, Mass.
- Thomas, A., Ventura, S., Jensen, S., and Ma, S. (2013). Competing process hazard function models for player ratings in ice hockey. *The Annals of Applied Statistics*, 7(3):1497–1524.

---

# Learning Latent Variable Models by Improving Spectral Solutions with Exterior Point Method

---

Amirreza Shaban    Mehrdad Farajtabar    Bo Xie    Le Song    Byron Boots  
College of Computing,  
Georgia Institute of Technology  
{amirreza, mehrdad, bo.xie}@gatech.edu    {lsong, boots}@cc.gatech.edu

## Abstract

Probabilistic latent-variable models are a fundamental tool in statistics and machine learning. Despite their widespread use, identifying the parameters of basic latent variable models continues to be an extremely challenging problem. Traditional maximum likelihood-based learning algorithms find valid parameters, but suffer from high computational cost, slow convergence, and local optima. In contrast, recently developed spectral algorithms are computationally efficient and provide strong statistical guarantees, but are not guaranteed to find valid parameters. In this work, we introduce a two-stage learning algorithm for latent variable models. We first use a spectral method of moments algorithm to find a solution that is *close* to the optimal solution but not necessarily in the valid set of model parameters. We then incrementally refine the solution via an exterior point method until a local optima that is arbitrarily near the valid set of parameters is found. We perform several experiments on synthetic and real-world data and show that our approach is more accurate than previous work, especially when training data is limited.

## 1 INTRODUCTION & RELATED WORK

Probabilistic latent variable models are a fundamental tool in statistics and machine learning that have successfully been deployed in a wide range of applied domains including robotics, bioinformatics, speech recognition, document analysis, social network modeling, and economics. Despite their widespread use, identifying the parameters of basic latent variable models like multi-view models and hidden Markov models (HMMs) continues to be an extremely challenging problem. Researchers often resort to local search heuristics such as expectation maximization (EM) (Dempster et al., 1977) that attempt to find param-

eters that maximize the likelihood of the observed data. Unfortunately, EM has a number of well-documented drawbacks, including high computational cost, slow convergence, and local optima.

In the past 5 years, several techniques based on *method of moments* (Pearson, 1894) have been proposed as an alternative to maximum likelihood for learning latent variable models (Hsu et al., 2009; Siddiqi et al., 2010; Song et al., 2010; Parikh et al., 2011, 2012; Hsu and Kakade, 2012; Anandkumar et al., 2012a,c,b; Balle et al., 2012; Cohen et al., 2013; Song et al., 2014). These algorithms first estimate low-order moments of observations, such as means and covariances, and then apply a sequence of linear algebra to recover the model parameters. Moment estimation is linear in the number of training data samples, and parameter estimation, which relies on techniques like the singular value decomposition (SVD) is typically fast and numerically robust.

For example, moment-based algorithms have been proposed for learning *observable* representations of HMMs, which explicitly avoid recovering HMM transition and observation matrices (Hsu et al., 2009; Siddiqi et al., 2010; Song et al., 2010). These *spectral* algorithms first perform a SVD of second-order moments of adjacent observations, and then use this result, along with additional low-order moments, to recover parameters for filtering, predicting, and simulating from the system. Unlike previous maximum likelihood-based approaches, spectral algorithms are fast, statistically consistent, and do not resort to local search.

Spectral algorithms were recently extended to the more difficult problem of estimating the parameters of latent variable models including the stochastic transition and observation matrices of HMMs (Anandkumar et al., 2012c).<sup>1</sup> Again, the estimators are based on SVD and a sequence of linear operations, applied to low-order moments of observations and come with learning guarantees under mild rank

---

<sup>1</sup>In contrast to the *observable* representation identified by the previous spectral learning algorithms (Hsu et al., 2009; Siddiqi et al., 2010; Song et al., 2010).

conditions. This work has been further extended to learning parameters of parametric and nonparametric multi-view latent variable models (Anandkumar et al., 2012b; Song et al., 2014) by introducing a symmetric tensor decomposition algorithm that unifies several previous method of moments-based approaches.

One of the benefits of method of moments over EM and other local search heuristics is that moment-based algorithms come with theoretical guarantees such as statistical consistency and finite sample bounds. In other words, under mild assumptions, method of moments can guarantee that as the amount of training data increases, the learned parameters are converging to the true parameters of the model that generated the data (Hsu et al., 2009; Anandkumar et al., 2012b). This is especially promising because the resulting parameters can be used to *initialize* EM in a two-stage learning algorithm (Zhang et al., 2014; Balle et al., 2014), resulting in the best of both worlds: parameters found by method of moments provide a good initialization for a maximum likelihood approach.

Unfortunately, spectral method of moments algorithms and two-stage learning algorithms have worked less well in practice. With finite samples, method of moments estimators are *not* guaranteed to find a valid set of parameters. Although error in the estimated parameters are bounded, the parameters themselves may lie outside the class of valid models. For example, the learned transition matrix of a HMM may have small negative entries. A consequence is that the learned model cannot be used or even serve as an initialization for EM.

To fix these problems, the method of moments solution is typically projected onto the space of valid model parameters: e.g. by flipping the sign of negative parameters and renormalizing the model (Cohen et al., 2013), or projecting the parameters onto the  $\ell_1$ -ball (Duchi et al., 2008). While these heuristics produce a useable model, it invalidates any theoretical guarantees: the resulting model may no longer be close to the true parameters. As demonstrated in Balle et al., models that are learned by method of moments and then “corrected” in this way, do not necessarily serve as a good initialization to EM (Balle et al., 2014).

### 1.1 EXTERIOR POINT METHODS

Consider the problem of minimizing objective function  $r(\mathbf{v}) : \mathbb{R}^n \rightarrow \mathbb{R}^+$ , subject to the constraint  $\mathbf{v} \in \mathcal{A}$ . Generally, a series of unconstrained optimization problem are solved to achieve a local optima in the limit. The optimization problem in the  $k^{th}$  step can be written as:

$$\text{minimize } r(\mathbf{v}) + l_k(\mathbf{v})$$

with the local optima  $\mathbf{v}^{(k)}$ . By defining the function  $l_k(\mathbf{v})$  appropriately, one can then show that  $\mathbf{v}^* = \lim_{k \rightarrow \infty} \mathbf{v}^{(k)}$  is a local optima of the original constrained optimization

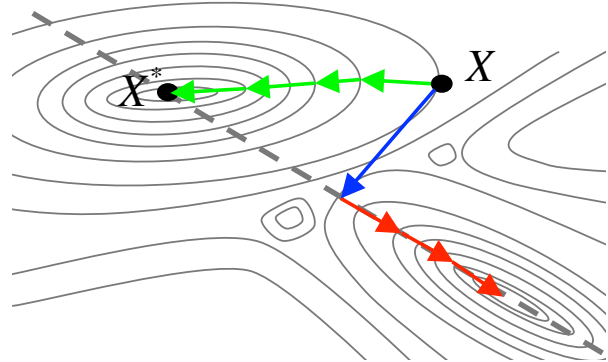


Figure 1: Exterior point methods versus projection followed by interior point methods. The dashed line shows the feasible set and the optimal solution is labeled  $\mathbf{X}^*$ . Starting from the method of moments solution  $\mathbf{X}$ , the exterior point method (green arrows) converges to a point arbitrarily close to the feasible set. Current optimization methods for learning latent variable models first project the method of moments solution into the feasible set (blue arrow) and then use an interior point method (red arrows show interior point method trajectory). Assuming that the initial solution is near to the optimal solution, as in this example, the projection step may change the convergence point to a point far from the optimal solution  $\mathbf{X}^*$ .

problem (Bloom, 2014). In *Interior point* methods every intermediate solution  $\mathbf{v}^{(k)}$  is in the feasible set, however, in *exterior point* methods, only the convergence point of the sequence needs to be feasible (Byrne, 2008). Examples of interior and exterior point methods are Barrier function methods (Boyd and Vandenberghe, 2004), and exact penalty function methods (Fletcher, 2013) respectively. In exterior point methods, the function  $l_k(\mathbf{v})$  usually has a positive value for solutions outside the feasible set to discourage these solutions and a value of zero for feasible inputs. In interior point methods, the function  $l_k(\mathbf{v}) \rightarrow \infty$  when  $\mathbf{v}$  approaches to the boundary of the constraint set. Polyak (2008), and Yamashita and Tanabe (2010) propose primal-dual exterior point methods for convex and non-convex optimization problems respectively. In Section 3, we show that by defining  $l_k(\mathbf{v})$  appropriately, the algorithm converges to a local optima arbitrarily close to the feasible set by doing simple *forward-backward splitting* steps (Combettes and Pesquet, 2011).

An important advantage of exterior point methods is that they are likely to achieve a better local minimum than interior point methods when the feasible set is narrow by allowing solutions to exist outside the feasible set during the intermediate steps of the optimization (Yamashita and Tanabe, 2010).

### 1.2 THE PROPOSED METHOD

One of the primary drawbacks of using method of moments for learning latent variable models, is that the estimated pa-

parameters can lie outside the class of valid models (Balle et al., 2014). To combat this problem, we propose a two-stage algorithm for learning the parameters of latent variable models.

In the first stage, the parameters are estimated via a spectral method of moments algorithm (similar to Anandkumar et al. (2012c)). Like previous method of moments-based learning algorithms, if the estimated moments are inaccurate, then the estimated parameters of this model may lie outside of the model class.

In the second stage, the estimate is refined by an iterative optimization scheme. Unlike previous work that projects method of moments onto the feasible space of model parameters and then uses the projected parameters to initialize EM (Zhang et al., 2014; Balle et al., 2014), we use exterior point methods for non-convex optimization directly initialized with the result of method of moments *without modification*. The exterior point method iteratively refines the solution until a local optima that is arbitrarily close to the valid set of model parameters is found. A comparison between the two approaches is illustrated in Figure 1.

### 1.3 BASICS AND NOTATION

We use following notation to distinguish scalars, vectors, matrices, and third-order tensors: scalars are denoted by either lowercase or uppercase letters, vectors are written as boldface lowercase letters, matrices correspond to boldface uppercase letters, and third-order tensors are represented by calligraphic letters. In this paper,  $(\mathbf{A})_{ij}$  means the entry in the  $i^{\text{th}}$  row and  $j^{\text{th}}$  column of the matrix  $\mathbf{A}$ , we use similar notation to index entries of vectors and third-order tensors. Furthermore, the  $i^{\text{th}}$  column of the matrix  $\mathbf{A}$  is denoted as  $(\mathbf{A})_i$ , i.e.,  $\mathbf{A} = [(\mathbf{A})_1, (\mathbf{A})_2, \dots, (\mathbf{A})_n]$ . We show a  $n \times m$  matrix with entries one by  $\mathbf{1}^{n \times m}$ ,  $n \times n$  identity matrices by  $\mathbf{I}^n$ , and  $n \times n \times n$  identity tensors by  $\mathcal{I}^n$ . We use  $\mathbb{R}_+^{n \times m}$  to show the set of  $n$  by  $m$  matrices with non-negative entries, and  $\Delta^n$  is the set of all  $n + 1$ -dimensional vector on the  $n$ -dimensional simplex.

We also define the following functions for ease of notation:  $\text{sum}(\mathbf{X}) = \mathbf{X}^\top \mathbf{1}$  computes column sum of the matrix  $\mathbf{X}$ , and the function  $\text{diag}(\mathbf{v})$ , which returns a diagonal matrix where its diagonal elements are a vector  $\mathbf{v}$ . For matrices or 3-way tensors,  $\text{diag}(\cdot)$  returns diagonal elements of the given input in vector form.

#### 1.3.1 $n$ -mode product (Lathauwer et al., 2000)

The  $n$ -mode product of a tensor  $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$  by a matrix  $\mathbf{B} \in \mathbb{R}^{J_n \times I_n}$  for  $1 \leq n \leq 3$ , shown as  $\mathcal{A} \times_n \mathbf{B}$ , is an  $K_1 \times K_2 \times K_3$  tensor, for which  $K_i = I_i$  for all dimensions except the  $n$ -th one which is  $K_n = J_n$ . The entries

are given by:

$$(\mathcal{A} \times_n \mathbf{B})_{i_1 \dots j_n \dots i_3} = \sum_{i_n} (\mathcal{A})_{i_1 \dots i_n \dots i_3} (\mathbf{B})_{j_n i_n}. \quad (1)$$

We benefit from following properties of  $n$ -mode product in future sections:

- Given a tensor  $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$  and a matrix  $\mathbf{C} \in \mathbb{R}^{J_n \times I_n}$  of the same size as  $\mathbf{B}$ , one can show that:

$$(\mathcal{A} \times_n \mathbf{B}) \times_n \mathbf{C} = \mathcal{A} \times_n (\mathbf{C}\mathbf{B}). \quad (2)$$

- For a matrix  $\mathbf{D} \in \mathbb{R}^{J_m \times I_m}$  ( $n \neq m$ ):

$$(\mathcal{A} \times_n \mathbf{B}) \times_m \mathbf{D} = (\mathcal{A} \times_m \mathbf{D}) \times_n \mathbf{B}.$$

- For matrices  $\mathbf{A}$ ,  $\mathbf{B}$ , and  $\mathbf{C}$  with appropriate sizes:

$$\mathbf{A} \times_1 \mathbf{B} \times_2 \mathbf{C} = \mathbf{B}\mathbf{A}\mathbf{C}^\top.$$

## 2 PARAMETER ESTIMATION VIA METHOD OF MOMENTS

In this section we derive two method of moments algorithms for estimating the parameters of latent variable models, one for multi-view models and one for HMMs. If the estimated parameters lie outside the feasible set of solutions, they are used to initiate an exterior point method (Section 3).

### 2.1 MULTI-VIEW MODELS

In a multi-view model, observation variables  $o_1, o_2, \dots, o_l$  are conditionally independent given a latent variable  $h$  (Figure 2a). Assume each observation variable can take one of  $n_o$  different values. The observation vector  $\mathbf{x}_t \in \mathbb{R}^{n_o}$  is defined as follows:

$$\mathbf{x}_t = \mathbf{e}_j \text{ iff } o_t = j \text{ for } 0 < j \leq n_o, \quad (3)$$

where  $\mathbf{e}_j$  is the  $j^{\text{th}}$  canonical basis. In this paper, we consider the case where  $l = 3$ , however, the techniques can be easily extended to cases where  $l > 3$ . Let  $h \in \{1, \dots, n_s\}$  be a discrete random variable and  $\Pr\{h = j\} = (\mathbf{w})_j$ , where  $\mathbf{w} \in \Delta^{n_s-1}$ , then the conditional expectation of observation vector  $\mathbf{x}_t$  for  $t \in \{1, 2, 3\}$  is:

$$\mathbb{E}[\mathbf{x}_t | h = i] = \mathbf{u}_i^t, \quad (4)$$

where  $\mathbf{u}_i^t \in \Delta^{n_o-1}$ . We define the observation matrix as  $\mathbf{U}^t = [\mathbf{u}_1^t, \dots, \mathbf{u}_{n_s}^t]$  for  $t \in \{1, 2, 3\}$ , and the diagonal  $n_s \times n_s \times n_s$  tensor  $\mathcal{H}$ , where  $\text{diag}(\mathcal{H}) = \mathbf{w}$  for ease of notation. The following proposition relates  $\mathbf{U}^t$ s and  $\mathbf{w}$  to the moments of  $\mathbf{x}_t$ s.

**Proposition 1. (Anandkumar et al., 2012b)** Assume that columns of  $\mathbf{U}^t$  are linearly independent for each  $t \in \{1, 2, 3\}$ . Define

$$\mathcal{M} = \mathbb{E}(\mathbf{x}_1 \otimes \mathbf{x}_2 \otimes \mathbf{x}_3)$$

Then

$$\mathcal{M} = \mathcal{H} \times_1 \mathbf{U}^1 \times_2 \mathbf{U}^2 \times_3 \mathbf{U}^3 \quad (5)$$

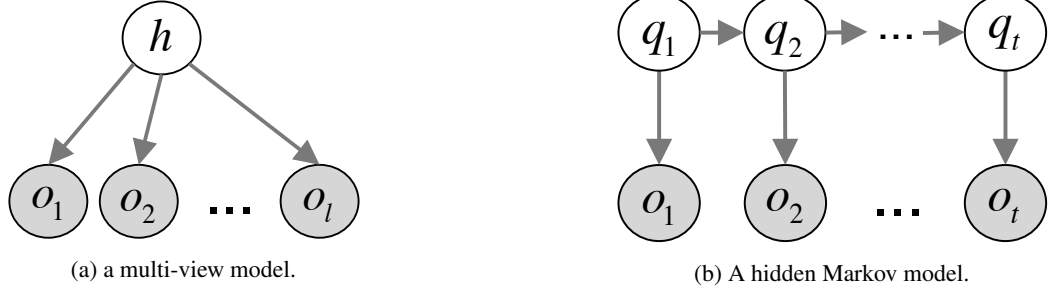


Figure 2: The two latent variable models discussed in the text.

In the next proposition, the moments of  $\mathbf{x}_t$ s are related to a specific  $\mathbf{U}^t$ :

**Proposition 2. (Anandkumar et al., 2012b)** *Assume that columns of  $\mathbf{U}_t$  are linearly independent for each  $t = \{1, 2, 3\}$ . Let  $(a, b, c)$  be a permutation of  $\{1, 2, 3\}$ . Define*

$$\begin{aligned} \mathbf{x}'_a &= \mathbb{E}(\mathbf{x}_c \otimes \mathbf{x}_b) \mathbb{E}(\mathbf{x}_a \otimes \mathbf{x}_b)^{-1} \mathbf{x}_a \\ \mathbf{x}'_b &= \mathbb{E}(\mathbf{x}_c \otimes \mathbf{x}_a) \mathbb{E}(\mathbf{x}_b \otimes \mathbf{x}_a)^{-1} \mathbf{x}_b \\ \mathbf{M}_c &= \mathbb{E}(\mathbf{x}'_a \otimes \mathbf{x}'_b) \\ \mathcal{M}_c &= \mathbb{E}(\mathbf{x}'_a \otimes \mathbf{x}'_b \otimes \mathbf{x}_c) \end{aligned}$$

Then

$$\begin{aligned} \mathbf{M}_c &= \mathbf{U}^c \text{diag}(\mathbf{w}) \mathbf{U}^{c\top} \\ \mathcal{M}_c &= \mathcal{H} \times_1 \mathbf{U}^c \times_2 \mathbf{U}^c \times_3 \mathbf{U}^c \end{aligned} \quad (6)$$

Also, define  $\mathbf{m}^c = \mathbb{E}(\mathbf{x}_c) = \mathbf{U}^c \mathbf{w}$ . Anandkumar et al. (2012b) transform  $\mathcal{M}_c$  to a orthogonally decomposable tensor and recover the matrices  $\mathbf{U}^t$ s and  $\mathbf{w}$  from it. Our approach here is slightly different and is more similar to Anandkumar et al. (2012c): we reduce the problem into the orthogonal decomposition of a matrix derived from  $\mathcal{M}_c$ .

First, let  $\mathbf{S} = \mathbf{V} \boldsymbol{\Sigma}^{-1/2}$ , where columns of  $\mathbf{V}$  are orthonormal eigenvectors of  $\mathbf{M}_c$  and  $\boldsymbol{\Sigma}$  is a diagonal matrix whose elements are corresponding eigenvalues of  $\mathbf{V}$ . The columns of  $\tilde{\mathbf{U}}^c = \mathbf{S}^\top \mathbf{U}^c \text{diag}(\mathbf{w})^{1/2}$  are orthonormal vectors (Anandkumar et al., 2012b). Using this and Equation (6) we have:

$$\begin{aligned} \mathbf{M}_\eta &= \mathcal{M}_c \times_1 \mathbf{S}^\top \times_2 \mathbf{S}^\top \times_3 \boldsymbol{\eta}^\top \\ &= \mathcal{H} \times_1 (\mathbf{S}^\top \mathbf{U}^c) \times_2 (\mathbf{S}^\top \mathbf{U}^c) \times_3 (\boldsymbol{\eta}^\top \mathbf{U}^c) \\ &= \mathcal{I}_{n_s} \times_1 \tilde{\mathbf{U}}^c \times_2 \tilde{\mathbf{U}}^c \times_3 (\boldsymbol{\eta}^\top \mathbf{U}^c) \\ &= \tilde{\mathbf{U}}^c \text{diag}(\boldsymbol{\eta}^\top \mathbf{U}^c) (\tilde{\mathbf{U}}^c)^\top \end{aligned} \quad (7)$$

where  $\boldsymbol{\eta}$  is a random vector sampled from the  $n_o$  dimensional normal distribution. For the first equality we used property in Equation (2), in the second equality we used the fact that  $\mathcal{H} = \mathcal{I}_{n_s} \times_1 \text{diag}(\mathbf{w})^{1/2} \times_2 \text{diag}(\mathbf{w})^{1/2}$ , and finally in the last equality we used equality  $\mathcal{I}_{n_s} \times_3 (\boldsymbol{\eta}^\top \mathbf{U}^c) = \text{diag}(\boldsymbol{\eta}^\top \mathbf{U}^c)$ . In Anandkumar et al. (2012c) it is shown that  $\boldsymbol{\eta}^\top \mathbf{U}^c$  has distinct values with a high probability. Thus,  $\tilde{\mathbf{U}}^c$  can be recovered by a SVD decomposition of  $\mathbf{M}_\eta$ . Then  $\mathbf{w} = ((\tilde{\mathbf{U}}^c)^\top \mathbf{S}^\top \mathbf{m}^c)^\wedge 2$ ,

---

**Algorithm 1** *Moment-based parameter estimation*

---

**Input:** Estimated third order moment  $\hat{\mathcal{M}}_c$  for  $c = \{1, 2, 3\}$

**Output:** Estimated parameters  $\hat{\mathbf{U}}^1, \hat{\mathbf{U}}^2, \hat{\mathbf{U}}^3, \hat{\mathbf{w}}$

---

**for each**  $t \in \{1, 2, 3\}$  **do**

$\hat{\mathbf{M}}_t \leftarrow \hat{\mathcal{M}}_t \times_3 \mathbf{1}$  (compute second-order moment)

$\hat{\mathbf{m}}_t \leftarrow \hat{\mathcal{M}}_t \times_2 \mathbf{1} \times_3 \mathbf{1}$  (compute first-order moment)

$\mathbf{S} \leftarrow \mathbf{V} \boldsymbol{\Sigma}^{-1/2}$  ( $\mathbf{V} \boldsymbol{\Sigma} \mathbf{V}^\top$  is  $\hat{\mathbf{M}}_t$ 's eigenvalue decomposition)

$\boldsymbol{\eta} \leftarrow$  drawn randomly from Normal distribution

$\mathbf{M}_\eta \leftarrow \hat{\mathcal{M}}_t \times_1 \mathbf{S}^\top \times_2 \mathbf{S}^\top \times_3 \boldsymbol{\eta}^\top$  (Eq. 7)

$\tilde{\mathbf{U}}^t \leftarrow \mathbf{K}$  (columns of  $\mathbf{K}$  are  $\mathbf{M}_\eta$ 's eigenvectors)

$\hat{\mathbf{w}}^t \leftarrow ((\tilde{\mathbf{U}}^t)^\top \mathbf{S}^\top \hat{\mathbf{m}}^t)^\wedge 2$

$\hat{\mathbf{U}}^t \leftarrow \mathbf{S}^{+\top} \tilde{\mathbf{U}}^t \text{diag}(\hat{\mathbf{w}}^t)^{-1/2}$

**end for**

$\hat{\mathbf{w}} \leftarrow \frac{\hat{\mathbf{w}}^1 + \hat{\mathbf{w}}^2 + \hat{\mathbf{w}}^3}{3}$

---

where  $^\wedge$  is element-wise power operator. Having computed  $\mathbf{w}$ , one can recover  $\mathbf{U}^c$  via the equation  $\mathbf{U}^c = \mathbf{S}^{+\top} \tilde{\mathbf{U}}^c \text{diag}(\mathbf{w})^{-1/2}$ . Finally, we take the average of 3 copies of  $\mathbf{w}$  which are computed for different values of  $c$ . The overall moment-based approach is shown in Algorithm 1.

## 2.2 HIDDEN MARKOV MODELS

Hidden Markov models generate sequences of observations  $\mathbf{x}_1, \mathbf{x}_2, \dots \in \mathbb{R}^{n_o}$ . Each  $\mathbf{x}_t$  is independent of all other observations given the corresponding hidden state  $q_t \in \{1, 2, \dots, n_s\}$  (Figure 2b). Similar to multi-view models,  $n_s$  and  $n_o$  are the number of hidden states and number of observations respectively. Note that observations are represented as *indicator vectors*  $\mathbf{x}_t$ , which are all zero except for exactly one element which is set to 1. The conditional probability distribution of  $\mathbf{x}_t$  given  $q_t$  is defined via an observation matrix  $\mathbf{O} \in \mathbb{R}^{n_o \times n_s}$  according to  $\Pr\{\mathbf{x}_t = \mathbf{e}_i | q_t = j\} = (\mathbf{O})_{ij}$ . The stochastic transition matrix  $\mathbf{T} \in \mathbb{R}^{n_s \times n_s}$  is defined as  $\Pr\{q_{t+1} = i | q_t = j\} = (\mathbf{T})_{ij}$  for all  $t > 1$  and the initial state probability distribu-

tion is  $\boldsymbol{\pi} \in \Delta^{n_s-1}$ . If  $\mathfrak{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T)$  is a sequence of observations, then we define forward and backward variables as

$$\begin{aligned} \Pr\{\mathbf{x}_1, \dots, \mathbf{x}_t, q_t = j\} &= \alpha_t(j) \\ \Pr\{\mathbf{x}_{t+1}, \dots, \mathbf{x}_T, q_t = j\} &= \beta_t(j) \end{aligned} \quad (8)$$

These will help computing the probability of observations. For example,

$$f(\mathfrak{X}; [\mathbf{O}, \mathbf{T}, \boldsymbol{\pi}]) = \sum_{i=1}^{n_s} \sum_{j=1}^{n_s} \alpha_t(i) (\mathbf{T})_{ij} (\mathbf{O})_j^\top \mathbf{x}_t \beta_{t+1}(j) \quad (9)$$

for all  $1 \leq t \leq T$  (Levinson et al., 1983). Note that the values of function  $\alpha_t(\cdot)$  and  $\beta_t(\cdot)$  can be computed efficiently using dynamic programming.

Under mild conditions, HMM parameter estimation reduces to estimating multi-view model parameters, using considering triple of observation  $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$ .

**Proposition 3. (Anandkumar et al., 2012b)** *let  $h = q_2$  then:*

- $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$  are conditionally independent given  $h$ .
- The distribution of  $h$  is  $\mathbf{w} = \mathbf{T}\boldsymbol{\pi}$ .
- For all  $j \in \{1, 2, \dots, n_s\}$

$$\begin{aligned} \mathbb{E}[\mathbf{x}_1 | h = j] &= \mathbf{O} \text{diag}(\boldsymbol{\pi}) \mathbf{T}^\top \text{diag}(\mathbf{w})^{-1/2} \mathbf{e}_j \\ \mathbb{E}[\mathbf{x}_2 | h = j] &= \mathbf{O} \mathbf{e}_j \\ \mathbb{E}[\mathbf{x}_3 | h = j] &= \mathbf{O} \mathbf{T} \mathbf{e}_j \end{aligned}$$

under mild conditions.

Thus, provided that  $\mathbf{O}$  and  $\mathbf{T}$  both have full column rank, the parameters of HMM can be recovered as  $\mathbf{O} = \mathbf{U}^2$ ,  $\mathbf{T} = \mathbf{O}^+ \mathbf{U}^3$ , and  $\boldsymbol{\pi} = \mathbf{T}^{-1} \mathbf{w}$ .

It is also important to note that, using the above proposition and Equation (9) we can alternatively write each entries of tensor  $\mathcal{M} = \mathbb{E}(\mathbf{x}_1 \otimes \mathbf{x}_2 \otimes \mathbf{x}_3)$  as:

$$(\mathcal{M})_{ijk} = f((\mathbf{e}_i, \mathbf{e}_j, \mathbf{e}_k); [\mathbf{O}, \mathbf{T}, \boldsymbol{\pi}]). \quad (10)$$

### 3 EXTERIOR POINT METHODS

While exact parameters can be recovered from the population moments, in practice we work with empirical moments  $\hat{\mathcal{M}}$ , and  $\hat{\mathcal{M}}$  which are computed using a finite set of training data. Thus, the estimated parameters are not necessarily exact and do not necessarily minimize the estimation error  $\|\hat{\mathcal{M}} - \mathcal{H} \times_1 \mathbf{U}^1 \times_2 \mathbf{U}^2 \times_3 \mathbf{U}^3\|_F$ .

In this section, we show that estimated parameters from Section 2 can directly initialize an iterative exterior point method that minimizes the above error while obeying constraints on model parameters. Although this initial seed may violate these constraints, we show that under mild conditions the parameters satisfy the model constraints once

the algorithm converges. First, we prove the convergence of the algorithm for multi-view models and then show how the algorithm can be applied to HMMs.

### 3.1 MULTI-VIEW MODELS

Let  $\mathbf{v} \in \mathbb{R}^{n_s(3n_o+1)}$  be a vector comprised of the parameters of the multi-view model  $\{\mathbf{U}^1, \mathbf{U}^2, \mathbf{U}^3, \text{diag}(\mathcal{H})\}$ , and  $\mathcal{R}(\mathbf{v}) = (\hat{\mathcal{M}} - \mathcal{H} \times_1 \mathbf{U}^1 \times_2 \mathbf{U}^2 \times_3 \mathbf{U}^3)$  be the residual estimation tensor. For ease of notation, we also define function  $s(\cdot) : \mathbb{R}^{n_s(3n_o+1)} \rightarrow \mathbb{R}^{3n_s+1}$  that computes column sum of  $\mathbf{U}^1, \mathbf{U}^2, \mathbf{U}^3$ , and  $\text{diag}(\mathcal{H})$ . As discussed above, the estimated parameters in the previous section do not necessarily minimize the estimation error  $\|\mathcal{R}(\mathbf{v})\|_F$  and also may violate the constraints for the model parameters. With these limitations in mind, we rewrite the factorization in Equation (6) in the form of an optimization problem:

$$\begin{aligned} &\text{minimize } \frac{1}{2} \|\mathcal{R}(\mathbf{v})\|_F^2. \\ &\text{s.t. } \mathbf{v} \in \mathbb{R}_+^{n_s(3n_o+1)}, s(\mathbf{v}) = \mathbf{1} \end{aligned} \quad (11)$$

Defining optimization problem in this form has two advantages over maximum likelihood optimization schemes. First, since  $\hat{\mathcal{M}}$  is computed in the previous stage, the optimization cost is asymptotically independent of the number of training samples which makes the proposed optimization algorithm faster than EM for large training sets. Second, the value of this objective function  $\|\mathcal{R}(\mathbf{v})\|_F$  is also defined *outside* of the feasible set. We use this property to extend the optimization problem for a simple exterior point method in Section 3.1.1, below.

#### 3.1.1 The Optimization Algorithm

Instead of solving constrained optimization problem in Equation (11), we solve the following unconstrained optimization problem:

$$\begin{aligned} &\text{minimize } \frac{1}{2} \|\mathcal{R}(\mathbf{v})\|_F^2 + \frac{\lambda_1}{2} \|s(\mathbf{v}) - \mathbf{1}\|_p^2 \\ &\quad + \lambda_2 |\mathbf{v}|_-, \end{aligned} \quad (12)$$

where  $|\mathbf{v}|_-$  is the absolute sum of all negative elements in the vector  $\mathbf{v}$ , i.e.,  $|\mathbf{v}|_- = \sum_i |(\mathbf{v})_i|_-$ . We set  $p = 2$  in our method. For  $p = 1$ , there exists a  $\lambda_1$  and a  $\lambda_2$  such that the solution to this unconstrained optimization is *also* the solution to the objective in Equation (11). A thorough survey on solving non-differentiable exact penalty functions can be found in (Fletcher, 2013). Our approach, however, is different in the sense that for  $p = 2$  the solution of our optimization algorithm is not guaranteed to satisfy the constraints in Equation (11), however, we show in Theorem 7 that the solution will be arbitrarily close to the simplex. In return for this relaxation, the above optimization problem can be easily solved by a standard *forward-backward splitting* algorithm (Combettes and Pesquet, 2011). In this

---

**Algorithm 2** *The exterior point algorithm*


---

**Input:** Estimated third order moment  $\hat{\mathcal{M}}$ , initial point (obtained from Algorithm 1)  $\mathbf{v}^{(0)}$  is comprised of  $\{\hat{U}^1, \hat{U}^2, \hat{U}^3, \text{diag}(\hat{\mathcal{H}})\}$ , parameters  $\lambda_1$ , and  $\lambda_2$ , sequence  $\{\beta_k\}$ , and constant  $c > 0$   
**Output:** Convergence point  $\mathbf{v}^*$

---

```

k ← 0
while not converged do
  k ← k + 1
  if  $|\mathbf{v}^{(k-1)}|_- > 0$  then
     $\alpha_k \leftarrow \max\{c, \beta_k\}$ 
  else
     $\alpha_k \leftarrow \beta_k$ 
  end if
   $\tilde{\mathbf{v}}^{(k)} \leftarrow \mathbf{v}^{(k-1)} - \alpha_k \nabla g(\mathbf{v}^{(k-1)})$ 
   $\mathbf{v}^{(k)} \leftarrow \text{prox}(\tilde{\mathbf{v}}^{(k)})$  (see Equation (15))
end while

```

---

method, the function is split into a smooth part:

$$g(\mathbf{v}) = \frac{1}{2} \|\mathcal{R}(\mathbf{v})\|_F^2 + \frac{\lambda_1}{2} \|\mathbf{s}(\mathbf{v}) - \mathbf{1}\|_2^2 \quad (13)$$

and a non-smooth part  $\lambda_2 |\mathbf{v}|_-$ . We then minimize the objective function by alternating between a gradient step on the smooth part  $\nabla g(\mathbf{v})$  (forward) and proximal step of the non-smooth part (backward). The overall algorithm is shown in Algorithm 2 where  $\text{prox}(\tilde{\mathbf{v}}^{(k)})$  function is defined as

$$\mathbf{v}^{(k)} = \underset{\mathbf{y}}{\text{argmin}} (\alpha_k \lambda_2 |\mathbf{y}|_- + \frac{1}{2} \|\tilde{\mathbf{v}}^{(k)} - \mathbf{y}\|_F^2). \quad (14)$$

and optimized by following transformation (Shalev-Shwartz and Zhang, 2013):

$$(\mathbf{v}^{(k)})_i = \begin{cases} (\tilde{\mathbf{v}}^{(k)})_i + \alpha_k \lambda_2 & (\tilde{\mathbf{v}}^{(k)})_i < -\alpha_k \lambda_2 \\ 0 & -\alpha_k \lambda_2 \leq (\tilde{\mathbf{v}}^{(k)})_i < 0 \\ (\tilde{\mathbf{v}}^{(k)})_i & 0 \leq (\tilde{\mathbf{v}}^{(k)})_i \end{cases} \quad (15)$$

We only need to find the gradient of function  $g(\mathbf{v})$  for the given model parameter  $\mathbf{v}$ . Following lemma shows the gradient of  $g(\mathbf{v})$  can be computed efficiently.

**Lemma 4.** *Let  $r(\mathbf{v}) = \frac{1}{2} \|\mathcal{R}(\mathbf{v})\|_F^2$ . The following are true for all  $0 < i \leq n_s$ :*

- $\frac{\partial r(\mathbf{v})}{\partial \mathcal{H}_{iii}} = -\mathcal{R}(\mathbf{v}) \times_1 \mathbf{u}_i^1 \times_2 \mathbf{u}_i^2 \times_3 \mathbf{u}_i^3$
- $\nabla_{\mathbf{u}_i^1} r(\mathbf{v}) = -\mathcal{H}_{iii} \times \mathcal{R}(\mathbf{v}) \times_2 \mathbf{u}_i^2 \times_3 \mathbf{u}_i^3$
- $\nabla_{\mathbf{u}_i^2} r(\mathbf{v}) = -\mathcal{H}_{iii} \times \mathcal{R}(\mathbf{v}) \times_1 \mathbf{u}_i^1 \times_3 \mathbf{u}_i^3$
- $\nabla_{\mathbf{u}_i^3} r(\mathbf{v}) = -\mathcal{H}_{iii} \times \mathcal{R}(\mathbf{v}) \times_1 \mathbf{u}_i^1 \times_2 \mathbf{u}_i^2$

Next we show that by using the forward-backward splitting steps in Algorithm 2 there are lower bounds for  $\lambda_1$  and  $\lambda_2$  in which the iterative algorithm converges to a local optimum arbitrarily close to the simplex. For this purpose, we

assume that estimated parameters remain in a compact set during the optimization, then we use the following corollary to bound the gradient of function  $r(\mathbf{v})$ .

**Corollary 5.** *For every  $\mathbf{v}$  which is comprised of the multi-view parameters and is in the compact set  $\mathcal{F} = \{\mathbf{v} \mid \forall i, 0 < i \leq n_s, 1 < t \leq 3 : \|(\mathbf{U}^t)_i\|_2 \leq L, \|\text{diag}(\mathcal{H})\|_2 \leq L\}$ , every element of the gradient of the function  $r(\mathbf{v})$  is bounded as:*

$$\left| \frac{\partial r(\mathbf{v})}{\partial (\mathbf{v})_i} \right| \leq L^3 \|\mathcal{R}(\mathbf{v})\|_F \quad (16)$$

Although, the norm of the residual error can also be bounded by  $L$  in Equation (16), since we initiate the algorithm with method of moments estimation, its value remains considerably smaller than its upper bound during the iterative procedure in Algorithm 2. Let  $\Upsilon > \sup_k \|\mathcal{R}(\mathbf{v}^{(k)})\|_F$ ; the next lemma shows that for a large enough  $\lambda_2$  and after a fixed number of iterations, all of the elements in  $\mathbf{v}^{(k)}$  become non-negative.

**Lemma 6.** *Assuming that sequence  $\{\mathbf{v}^{(k)}\}$  produced by Algorithm 2 is in the set  $\mathcal{F}$ , and  $\lambda_2$  is selected such that:*

$$\lambda_2 > L^3 \Upsilon + \lambda_1 (\sqrt{n_o} L + 1), \quad (17)$$

*there is a constant  $K$  such that for  $k > K$  we have  $|\mathbf{v}^{(k)}|_- = 0$ . Also, for  $k > K$  the proximal operator in Algorithm 2 reduces to the orthogonal projection operator into the convex set  $\mathcal{C} = \mathbb{R}_+^{n_s(3n_o+1)}$ :*

$$\forall k > K : \text{prox}(\tilde{\mathbf{v}}^{(k)}) = \text{proj}_{\mathcal{C}}(\tilde{\mathbf{v}}^{(k)}) \quad (18)$$

The proof is provided in the Appendix. According to the above lemma, after  $K$  iterations of forward-backward splitting steps, all entries of  $\mathbf{v}^{(k)}$  have non-negative values and the optimization algorithm reduces to gradient projection steps (Bertsekas, 1999) into the set  $\mathbb{R}_+^{n_s(3n_o+1)}$  for optimizing non-convex function  $g(\mathbf{v})$ . There is a lot of research on the convergence guarantees of gradient projection methods for non-convex optimization with different line search algorithms (Bertsekas, 1999), which also can be used in Algorithm 2. The only requirement of our method is that stepsize  $\alpha_k$  should be strictly bounded away from zero for  $k \leq K$  which is guaranteed in Algorithm 2 by taking the  $\max\{c, \beta_k\}$  for  $k \leq K$  for an arbitrary constant  $c > 0$ . Finally, the following theorem shows that by choosing large enough  $\lambda_1$  and  $\lambda_2$ , the algorithm ends up with a solution arbitrarily close to the simplex.

**Theorem 7.** *For every  $\epsilon_1 > 0$ , set  $\lambda_1 > \frac{L^3 \Upsilon}{\epsilon_1}$  and  $\lambda_2 > L^3 \Upsilon + \lambda_1 (\sqrt{n_o} L + 1)$ , in Equation (12). For the convergence point of the sequence  $\{\mathbf{v}^{(k)}\} \subset \mathcal{F}$  which is generated by Algorithm 2 we have:*

$$|\mathbf{v}^*|_- = 0, \|\mathbf{s}(\mathbf{v}^*) - \mathbf{1}\|_2 \leq \epsilon_1$$

The proof of Theorem 7 is provided in the Appendix.

To summarize, we initiate our exterior point method with the result of the method of moments estimator in Sec-

tion 2.1. By choosing large enough  $\lambda_1$ , and  $\lambda_2$ , the convergence point of the exterior point method will be at a local optimum of  $g(\mathbf{v})$  with the constraint  $\mathbf{v} \in \mathbb{R}_+^{n_s(3n_o+1)}$  in which the column sum of parameters set is arbitrarily close to 1. It is important to note that the proven lower bounds for  $\lambda_1$ , and  $\lambda_2$  are sufficient condition for the convergence. In practice, cross-validation can find the best parameter to balance the speed of mapping to the simplex with optimizing the residual function.

### 3.2 HIDDEN MARKOV MODELS

In Section 2.2 we showed that method of moments parameter estimation for HMMs essentially reduces to parameter estimation of multi-view models. After finding parameters from the method of moments algorithm, Algorithm 2 can be used to further refine the solution. In order to use this algorithm, we just need to define the residual estimation term for HMMs, define the function  $g(\cdot)$ , and compute its gradient. Assuming the parameters of the HMM  $\mathbf{T}$ ,  $\mathbf{O}$ , and  $\boldsymbol{\pi}$  are as defined in Section 2.2, let vector  $\mathbf{z}$  be comprised of these parameters. Similar to the multi-view case when the estimated moments are not exact, the equality in Equation (10) does not hold, and we define the residual prediction error of the model for the triples  $(\mathbf{e}_i, \mathbf{e}_j, \mathbf{e}_k)$  as  $(\mathcal{R}(\mathbf{z}))_{ijk} = (\hat{\mathcal{M}})_{ijk} - f((\mathbf{e}_i, \mathbf{e}_j, \mathbf{e}_k); [\mathbf{O}, \mathbf{T}, \boldsymbol{\pi}])$ . Thus, the optimization problem is:

$$\text{minimize } g(\mathbf{z}) + \lambda_2 \|\mathbf{z}\|_-, \quad (19)$$

where  $g(\mathbf{z})$  is defined as:

$$g(\mathbf{z}) = \frac{1}{2} \|\mathcal{R}(\mathbf{z})\|_F^2 + \frac{\lambda_1}{2} \|\mathbf{s}(\mathbf{z}) - \mathbf{1}\|_2^2. \quad (20)$$

The following lemma shows that the gradient of the first term in above equation can be represented by forward and backward variables in Equation (8) efficiently.

**Lemma 8.** *Let  $r(\mathbf{z}) = \frac{1}{2} \|\mathcal{R}(\mathbf{z})\|_F^2$ , for all  $0 < a, b \leq n_s$  and  $0 < c \leq n_o$  following holds:*

- a.  $\frac{\partial r(\mathbf{z})}{\partial (\boldsymbol{\pi})_a} = \sum (\mathcal{R}(\mathbf{z}))_{ijk} (\mathbf{O})_a^\top \mathbf{x}_1 \beta_1(a)$
- b.  $\frac{\partial r(\mathbf{z})}{\partial (\mathbf{T})_{ab}} = \sum (\mathcal{R}(\mathbf{z}))_{ijk} \sum_{t=1}^2 \alpha_t(a) (\mathbf{O})_b^\top \mathbf{x}_{t+1} \beta_{t+1}(b)$
- c.  $\frac{\partial r(\mathbf{z})}{\partial (\mathbf{O})_{cb}} = \frac{1}{(\mathbf{O})_{cb}} \sum (\mathcal{R}(\mathbf{z}))_{ijk} \sum_{t: \mathbf{x}_t = \mathbf{e}_c} \alpha_t(b) \beta_t(b)$

where the outer sums are over  $0 < i, j, k < n_o$  and  $\mathbf{x}_1 = \mathbf{e}_i$ ,  $\mathbf{x}_2 = \mathbf{e}_j$ , and  $\mathbf{x}_3 = \mathbf{e}_k$  ( $\mathbf{e}_i$  is the  $i^{\text{th}}$  canonical basis).

To summarize: to estimate the parameters of a HMM, we initialize Algorithm 2 with the method of moments estimate of the parameters. Then, using lemma 8, we compute  $\nabla g(\mathbf{z})$  at each iteration to solve the optimization (Equation (19)).

## 4 EXPERIMENTAL RESULTS

We evaluate the performance of our proposed method (EX&SVD) on both synthetic and real world datasets. We compare our approach to several state-of-the-art alternatives including EM initialized with 10 random seeds (EM), EM initialized with the method of moments result described in Section 2 after projecting the estimated parameters into simplex (EM&SVD), and the recently published symmetric tensor decomposition method (STD) (Anandkumar et al., 2012b). To evaluate the performance gain due to exterior point algorithm, we also included results from method of moments without the additional optimization (SVD) Section 2.

To ensure a fair time comparison, all of the methods were implemented in Matlab. In all methods, the iteration was stopped whenever the change in  $\frac{obj^{(t-1)} - obj^{(t)}}{[avg(obj^{(t)}, obj^{(t-1)})]}$  was less than  $\delta$ . We set parameter  $\delta$  in EM-based approaches, and exterior point algorithm to  $10^{-4}$  (Murphy; Parikh et al., 2012), and  $10^{-3}$  respectively.

Parameters  $\lambda_1$ , and  $\lambda_2$  controls the speed of mapping parameters into the simplex while estimation error term is optimized simultaneously. We find the best parameters using cross-validation. In our experiments, we sample  $N$  training and  $M$  testing points from each model. For the evaluation we use  $M = 2000$  test samples and calculate normalized  $\ell_1$  error =  $\frac{1}{M} \sum_{i=1}^M \frac{|\mathbb{P}(X_i) - \hat{\mathbb{P}}(X_i)|}{\mathbb{P}(X_i)}$ .

We found that, empirically, spectral methods and exterior point algorithm outperform EM for small sample sizes. In these situations, we believe that EM is overfitting, resulting in poor performance on the test dataset. Similar results are also reported in (Parikh et al., 2012). As the number of training data points increases, EM begins to outperform the spectral methods. However, our experiments show that EX&SVD constantly outperforms other methods in terms of estimation error while remaining an order of magnitude faster than EM.

It is important to note that the gap between estimation error of EX&SVD and EM&SVD is considerably larger in the situations where the number of training data points is relatively small compared to the number of model parameters. In these situations, estimation error in the SVD method is not accurate and the error of projection into the simplex is relatively high in EM&SVD method. However, when the SVD parameter estimates are used to initialize our exterior point algorithm we get considerably better parameter estimates. When the SVD estimate of the parameters is accurate (due to the large training set and small number of parameters) EM&SVD and EX&SVD estimations are close to each other. We believe that this observation strongly supports our approach to use exterior point method to find a set of parameters in the valid set of models (rather than a naive projection).



## 4.1 MULTI-VIEW MODEL EXPERIMENTS

To test the performance of the proposed method on learning multi-view models in different settings, we generate observations from randomly sampled models. The first set of models has 5 hidden states and 10 discrete observations per view, and the second set of models has 10 hidden states and 20 observations per view.

Figure 3 shows the average error of the implemented algorithms run on 10 different datasets generated by *i.i.d* sampled models. Each dataset consisted of up to 100,000 triples of observations sampled from each model. We used log-log scale for better demonstration of results. In these experiments, EM is initialized with 10 different random seeds and the best model is reported. Both EM&SVD and EX&SVD are initialized with 1 sample from our SVD decomposition method. As discussed earlier, EM outperforms SVD and the tensor decomposition method with respect to estimation error as the number of training samples increases. However, both EX&SVD and EM&SVD outperform EM, which shows the effectiveness of using the method of moments result as an initial seed for optimization. The performance of the EX&SVD method is significantly better especially in the small sample size region where the method of moments result is far from the simplex and projection step in EM&SVD method change the value of initial seed a lot. As the number of training samples increases, the error induced by the projection decreases and the results of the two methods converge.

Comparing the results from the two model classes, we see that the difference between the performance of EX&SVD and other methods is more pronounced as the number of parameters increases. This is due to the fact that the error of SVD increases as the number of parameters increases; which, in turn, is due to poor population estimates of the moments. The error of projecting the SVD result into the simplex is also high in the EM&SVD method. As illustrated in Figure 3, the result of SVD is comparable to STD while SVD is orders of magnitude faster. Considering the large number of parameters in this experiment, it is not strange that both method of moments algorithms (STD and SVD) do not show a good performance, however, both EM&SVD and EX&SVD outperform EM which shows the method of moments estimation are a better initialization point than a random selection.

To study the performance of different methods under different parameter set sizes in more detail, we investigate the performance of the different algorithms in estimating parameters of models with different numbers of hidden states. To this end, we sample  $N = 4000$  training points from models with different numbers of hidden states and evaluate the performance of different method in estimating these models parameters. The average error of 10 independent runs is reported in figure 4 for different values of  $n_s$ . In

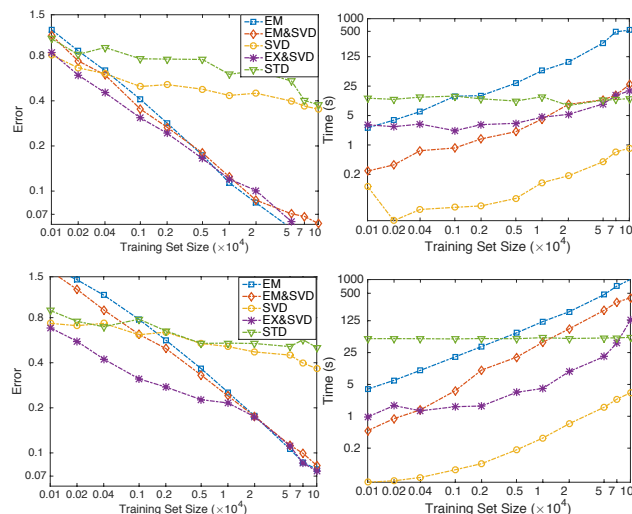


Figure 3: Error vs. #training (first column), and Time vs. #training (second column) for multi-view models.  $n_s = 5$ ,  $n_o = 10$  in the first row, and  $n_s = 10$ ,  $n_o = 20$  in the second row.

each case we set  $n_o$  to twice the value of  $n_s$ . As  $n_s$  increases, the number of model parameters also increases while the number of training points remains fixed. This results in the estimation error increasing as the models get larger for all of the methods. However, the difference between the performance of EX&SVD and other methods becomes more pronounced with  $n_s$ , which shows the effectiveness of our method in learning models with large state spaces and relatively smaller datasets.

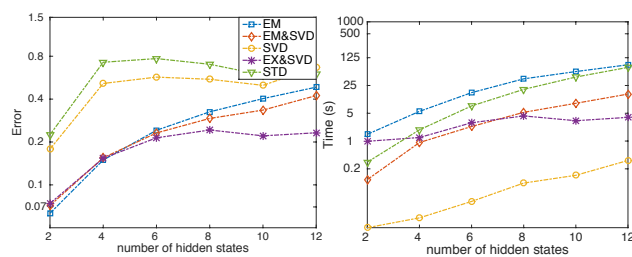


Figure 4: Error vs.  $n_s$  (left), time vs.  $n_s$  (right) for multi-view model for #training = 4000.

## 4.2 HIDDEN MARKOV MODEL EXPERIMENTS

We also evaluate the performance of our algorithm by estimating the parameters of HMMs on synthetic and real-world datasets. Similar to the multi-view case, we randomly sample parameters from two different classes of models to generate synthetic datasets. The first set of models again has 5 hidden states and 10 discrete observations, and the second set of models has 10 hidden states and 20 observations. Figure 5 shows the average error of the implemented algorithms run on 10 different datasets generated by *i.i.d* sampled models. Each dataset consisted of

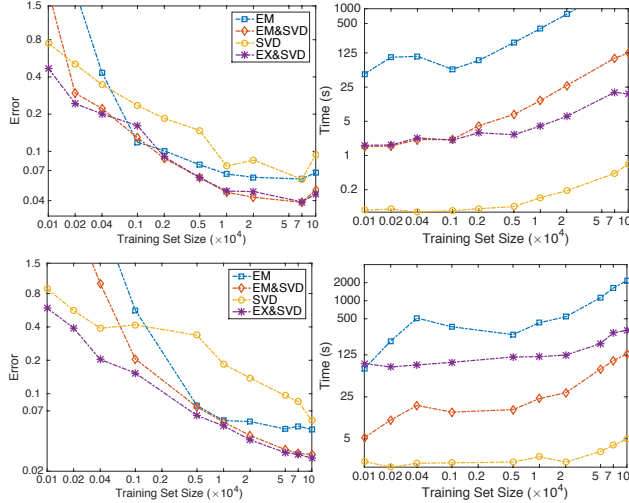


Figure 5: Error vs. #training (first column), and time vs. #training (second column) for HMM models.  $n_s = 5$ ,  $n_o = 10$  (first row), and  $n_s = 10$ ,  $n_o = 20$  (second row).

up to 100,000 triples of observations sampled from each model. Although, estimated parameters in the SVD method do not have good performance in terms of normalized  $l_1$  error, using them to initiate an iterative optimization procedure improves performance as demonstrated by both the EM&SVD and the EX&SVD methods. On the other hand, our algorithm can also outperform EM&SVD, especially in the low and medium sample size regions when the error of projection step is relatively high. For medium and large training set sizes, EM&SVD and EX&SVD are almost the same speed, and both are considerably faster than EM alone.

#### 4.2.1 Splice Dataset

In this experiment we consider the task of recognizing splice sites on a DNA sequence (Bache and Lichman, 2013). The dataset consist of 3190 examples. Each example is a sequence of 60 fields in which every field is filled by either A,T,C, or G. The label of each example could be Intron/Exon site, Exon/Intron site, or neither. For training, we train a HMM with  $n_s = 4$  for each class using different methods and use the rest of examples for testing. For each test example we compute the probability of the sequence for each model, and choose the label corresponding to the model with the highest test probability. For each test example in our method, we compute the histogram of triples in the test sequence, and choose the label corresponding to the model whose probability distribution over different triples has the lowest  $\ell_1$  distance to the computed histogram. This method of classification is a natural fit for our method, since our optimization method finds model parameters such that its probability distribution over different triples has minimum distance to the empirically estimated distribution of tripes  $\hat{\mathcal{M}}$ . Figure 6 shows the average classification error of each method for 10 randomly

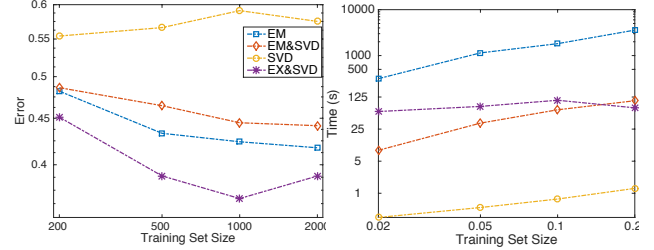


Figure 6: Error vs. #training (left), time vs. #training (right) for splice dataset.

chosen training set with several different sized training sets. The results are consistent with the experiments on the synthetic datasets in terms of speed and accuracy, despite EM outperforms EM&SVD in real world dataset. However, our method performs considerably better than EM.

## 5 CONCLUSION

We present a new approach to learning latent variable models such as multi-view models and HMMs. Recent work on learning such models by method of moments has produced exciting theoretical results that explicitly bound the error in the learned model parameters. Unfortunately, these results have failed to translate into accurate and numerically robust algorithms in practice. In particular, the parameters learned by method of moments may lie outside of the feasible set of models. This is especially likely to happen when the population moments are estimated inaccurately from small quantities of training data. To overcome this problem, we propose a two-stage algorithm for learning the parameters of latent variable models. In the first stage, we learn an initial estimate of the parameters by method of moments. In the second stage, we use an exterior point algorithm that incrementally refines the solution until the parameters are at a local optima and arbitrarily close to valid model parameters. We prove convergence of the method and perform several experiments to compare our method to previous work. An empirical evaluation on both synthetic and real-world datasets demonstrates that our algorithm learns models that are generally more accurate than method of moments or EM alone. By elegantly contending with parameters that may be outside of the model class, we are able to learn models that are much more accurate than EM initialized with method of moments when only a limited amount of training data is available.

## Acknowledgement

The research was supported in part by NSF IIS-1116886, NSF/NIH BIGDATA 1R01GM108341, and NSF CAREER IIS-1350983.

## References

- A. Anandkumar, D. P. Foster, D. Hsu, S. M. Kakade, and Y. Liu. Two svds suffice: Spectral decompositions for probabilistic topic modeling and latent dirichlet allocation. *CoRR*, abs/1204.6703, 2012a.
- A. Anandkumar, R. Ge, D. Hsu, S. M. Kakade, and M. Tegarsky. Tensor decompositions for learning latent variable models. *arXiv preprint arXiv:1210.7559*, 2012b.
- A. Anandkumar, D. Hsu, and S. M. Kakade. A method of moments for mixture models and hidden markov models. In *Proc. Annual Conf. Computational Learning Theory*, pages 33.1–33.34, 2012c.
- K. Bache and M. Lichman. UCI machine learning repository, 2013. URL <http://archive.ics.uci.edu/ml>.
- B. Balle, A. Quattoni, and X. Carreras. Local loss optimization in operator models: A new insight into spectral learning. In *Proceedings of the International Conference on Machine Learning*, 2012.
- B. Balle, W. Hamilton, and J. Pineau. Methods of moments for learning stochastic languages: Unified presentation and empirical comparison. In *Proceedings of the International Conference on Machine Learning*, pages 1386–1394, 2014.
- D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA, second edition, 1999.
- V. J. Bloom. *Exterior-Point Algorithms for Solving Large-Scale Nonlinear Optimization Problems*. PhD thesis, George Mason University, 2014.
- Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, England, 2004.
- C. Byrne. Sequential unconstrained minimization algorithms for constrained optimization. *Inverse Problems*, 24(1), 2008.
- S. B. Cohen, K. Stratos, M. Collins, D. P. Foster, and L. Ungar. Experiments with spectral learning of latent-variable PCFGs. In *Proceedings of NAACL*, 2013.
- P. L. Combettes and J. Pesquet. Proximal splitting methods in signal processing. In *Fixed-point algorithms for inverse problems in science and engineering*, pages 185–212. Springer, 2011.
- A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39(1):1–22, 1977.
- J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandrea. Efficient projections onto the  $\ell_1$ -ball for learning in high dimensions. In *Proceedings of the International Conference on Machine Learning*, 2008.
- R. Fletcher. *Practical methods of optimization*. John Wiley & Sons, 2013.
- D. Hsu and S.M. Kakade. Learning mixtures of spherical gaussians: moment methods and spectral decompositions, 2012. URL [arXiv:1206.5766](https://arxiv.org/abs/1206.5766).
- D. Hsu, S. Kakade, and T. Zhang. A spectral algorithm for learning hidden markov models. In *Proc. Annual Conf. Computational Learning Theory*, 2009.
- L. D. Lathauwer, B. D. Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000.
- S. E. Levinson, L. R. Rabiner, and M. M. Sondhi. An introduction to the application of the theory of probabilistic functions of a Markov process to automatic speech recognition. *Bell System Technical Journal*, 62(4):1035–1074, April 1983.
- K. Murphy. Hidden markov model (hmm) toolbox for matlab. URL <https://github.com/probml/pmtk3>.
- A. Parikh, L. Song, and E. P. Xing. A spectral algorithm for latent tree graphical models. In *Proceedings of the International Conference on Machine Learning*, 2011.
- A. P. Parikh, L. Song, M. Ishteva, G. Teodoru, and E.P. Xing. A spectral algorithm for latent junction trees. In *Conference on Uncertainty in Artificial Intelligence*, 2012.
- K. Pearson. Contributions to the mathematical theory of evolution. *Transactions of the Royal Society of London*, 185:71–110, 1894.
- R. A Polyak. Primal–dual exterior point method for convex optimization. *Optimisation Methods and Software*, 23(1):141–160, 2008.
- S. Shalev-Shwartz and T. Zhang. Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. *Mathematical Programming*, pages 1–41, 2013.
- S. Siddiqi, B. Boots, and G. J. Gordon. Reduced-rank hidden Markov models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS-2010)*, 2010.
- L. Song, B. Boots, S. Siddiqi, G. Gordon, and A. J. Smola. Hilbert space embeddings of hidden markov models. In *International Conference on Machine Learning*, 2010.
- L. Song, A. Anamdakumar, B. Dai, and B. Xie. Non-parametric estimation of multi-view latent variable models. In *International Conference on Machine Learning (ICML)*, 2014.
- H. Yamashita and T. Tanabe. A primal-dual exterior point method for nonlinear optimization. *SIAM Journal on Optimization*, 20(6):3335–3363, 2010.
- Y. Zhang, X. Chen, D. Zhou, and M. I Jordan. Spectral methods meet em: A provably optimal algorithm for crowdsourcing. In *Advances in Neural Information Processing Systems 14*, pages 1260–1268, 2014.

---

# Missing Data as a Causal and Probabilistic Problem

---

**Ilya Shpitser**

Mathematical Sciences  
University of Southampton  
Southampton, UK SO14 6WD  
i.shpitser@soton.ac.uk

**Karthika Mohan**

Dept. of Computer Science  
Univ. of California, Los Angeles  
Los Angeles, CA 90095  
karthika@cs.ucla.edu

**Judea Pearl**

Dept. of Computer Science  
Univ. of California, Los Angeles  
Los Angeles, CA 90095  
judea@cs.ucla.edu

## Abstract

Causal inference is often phrased as a missing data problem – for every unit, only the response to observed treatment assignment is known, the response to other treatment assignments is not. In this paper, we extend the converse approach of [7] of representing missing data problems to causal models where only interventions on missingness indicators are allowed. We further use this representation to leverage techniques developed for the problem of identification of causal effects to give a general criterion for cases where a joint distribution containing missing variables can be recovered from data actually observed, given assumptions on missingness mechanisms. This criterion is significantly more general than the commonly used “missing at random” (MAR) criterion, and generalizes past work which also exploits a graphical representation of missingness. In fact, the relationship of our criterion to MAR is not unlike the relationship between the **ID** algorithm for identification of causal effects [22, 18], and conditional ignorability [13].

## 1 INTRODUCTION

Missing data is a ubiquitous problem in data analysis, and can arise due to imperfect data collection, or various types of censoring, for instance via loss to followup, or death. In addition, causal inference can be viewed as a missing data problem, since the fundamental problem of causal inference [4] is that for every unit only the response to observed treatment assignment is known, the responses to other, hypothetical treatment assignments are not known.

Handling missing data entails either dealing with a latent variable model or finding plausible assumptions under which *recoverability*, that is unbiased inferences about *all* cases from the *observed* cases, is possible. Well-known approaches of the former type include fitting a latent variable

model via gradient descent [17], the EM algorithm [1], or performing Monte Carlo averaging via multiple imputation [16]. Well-known approaches of the latter type include the Kaplan-Meier estimator in survival analysis [5], and adjustments based on Missing Completely At Random (MCAR), and Missing At Random (MAR) assumptions [15].

While methods based on inference in a latent variable model are more generally applicable, they are also methodologically and computationally challenging. At the same time, recoverability methods based on MCAR and MAR rely on strong assumptions on how missingness comes about. When neither MCAR nor MAR holds, data is said to be Missing Not At Random (MNAR), and in this case a characterization of recoverability is an open problem, although many sufficient conditions for recoverability are known [7, 6].

In this paper, we take the *converse* view to “causality as missing data,” and view missing data as a particular type of partly causal, and partly probabilistic inference problem [2, 7]. We then represent this problem using partly causal, and partly probabilistic graphical models, and exploit techniques developed for similar models in the context of identification of causal effects to develop a general algorithm for recoverability under MNAR. In fact, the relationship between our algorithm and MAR is not unlike the relationship between the **ID** algorithm for identification of causal effects [22, 18, 19], and the conditional ignorability assumption in causal inference [13].

The paper is organized as follows. We introduce the notation and concepts we will need in section 2. In section 3, we use missingness graphs and missingness models to formally define missing data as a type of causal inference problem where only interventions on certain variables are allowed. We introduce recoverability and give examples of where recoverability is possible in MNAR settings in section 4. We introduce a general algorithm for recoverability we call **MID** in section 5, and show it is sound. Section 6 illustrates a complex case where the entire recursive structure of **MID** is necessary. Section 7 discusses non-recoverability, and section 8 contains our conclusions.

## 2 PRELIMINARIES

Variables are capital letters, values are small letters. Variable sets are bold capital letters, value sets are bold small letters. A state space for a variable  $A$  is  $\mathfrak{X}_A$ . A state space for a set of variables  $\mathbf{A}$  is the Cartesian product of the individual state spaces:  $\mathfrak{X}_{\mathbf{A}} \equiv \times_{A \in \mathbf{A}} \mathfrak{X}_A$ . For a set of values  $\mathbf{a}$ , and  $\mathbf{B} \subseteq \mathbf{A}$ , denote by  $\mathbf{a}_{\mathbf{B}}$  a projection of  $\mathbf{a}$  to  $\mathbf{B}$ . Denote  $\mathbf{a}_{\mathbf{B}}$  as a shorthand for  $\mathbf{a}_{\{\mathbf{B}\}}$ . We will denote a vector of 0s as  $\mathbf{0}$ .  $\mathbf{0}_{\mathbf{B}}$  means “a set of 0 values to  $\mathbf{B}$ .”

### 2.1 GRAPH THEORY AND NOTATION

A directed graph consists of a set of nodes and directed arrows ( $\rightarrow$ ) connecting pairs of nodes. A mixed graph consists of a set of nodes and directed and/or bidirected arrows ( $\leftrightarrow$ ) connecting pairs of nodes. A path is a sequence of distinct edges where any edge in a sequence that ends in a node  $A$  implies the subsequent edge must start with  $A$ , and each such node  $A$  may only occur at most once in this way in the sequence. A directed path from a node  $X$  to a node  $Y$  is a path consisting of directed edges where all edges on the path point away from  $X$  and towards  $Y$ .

If the edge  $X \rightarrow Y$  exists in a graph  $\mathcal{G}$ , we say  $X$  is a parent of  $Y$  and  $Y$  is a child of  $X$ . If a directed path from  $X$  to  $Y$  exists in  $\mathcal{G}$ , we say  $X$  is an ancestor of  $Y$ , and  $Y$  is a descendant of  $X$ . We denote by  $\text{pa}_{\mathcal{G}}(A)$ ,  $\text{ch}_{\mathcal{G}}(A)$ ,  $\text{de}_{\mathcal{G}}(A)$ ,  $\text{an}_{\mathcal{G}}(A)$ ,  $\text{nd}_{\mathcal{G}}(A)$  the sets of parents, children, descendants, ancestors, and non-descendants of  $A$  in  $\mathcal{G}$ , respectively. These are defined disjunctively for sets, e.g.  $\text{pa}_{\mathcal{G}}(\mathbf{A}) = \bigcup_{A \in \mathbf{A}} \text{pa}_{\mathcal{G}}(A)$ . Let  $\text{fa}_{\mathcal{G}}(A) = \text{pa}_{\mathcal{G}}(A) \cup \{A\}$ ,  $\text{pa}_{\mathcal{G}}^s(\mathbf{A}) = \text{pa}_{\mathcal{G}}(\mathbf{A}) \setminus \mathbf{A}$ ,  $\text{ndp}_{\mathcal{G}}(A) = \text{nd}_{\mathcal{G}}(A) \setminus \text{pa}_{\mathcal{G}}(A)$ . Given a graph  $\mathcal{G}$ , we say a vertex set  $\mathbf{A}$  is *ancestral* if  $\text{an}_{\mathcal{G}}(\mathbf{A}) = \mathbf{A}$ . By convention, in any directed graph,  $A \in \text{an}_{\mathcal{G}}(A) \cap \text{de}_{\mathcal{G}}(A)$ . A directed graph is said to have a directed cycle if there is  $X, Y$  such that  $X \in \text{an}_{\mathcal{G}}(Y) \cap \text{ch}_{\mathcal{G}}(Y)$ . A directed graph without such cycles is called a directed acyclic graph (DAG).

A *conditional DAG* (CDAG)  $\mathcal{G}(\mathbf{V} \mid \mathbf{W})$  is a DAG with vertices  $\mathbf{V} \cup \mathbf{W}$  with the property that  $\text{pa}_{\mathcal{G}}(\mathbf{W}) = \emptyset$ . We will denote vertices in  $\mathbf{V}$  as circles, and vertices in  $\mathbf{W}$  as squares. Note that we do not require that all  $V \in \mathbf{V}$  must have parents. We simply distinguish certain parentless nodes in  $\mathcal{G}$  as  $\mathbf{W}$ . We will interpret vertices in  $\mathbf{V}$  as associated with *random variables* and vertices in  $\mathbf{W}$  as associated with variables that have been “set to a constant” in some way. One example of a CDAG is a *mutilated graph* that arises in the analysis of interventional distributions. When considering d-separation on vertices in  $\mathbf{V}$  in a CDAG [9], we will treat it as ordinary d-separation in a DAG, except all nodes in  $\mathbf{W}$  are implicitly conditioned on.

If vertices not in  $\mathbf{W}$  in a CDAG correspond to a variable partition into observed and missing variables, we will explicitly denote the set of vertices corresponding to miss-

ing variables as  $\mathbf{M}$ , and the other vertices as  $\mathbf{O}$ , like so:  $\mathcal{G}(\mathbf{O}, \mathbf{M} \mid \mathbf{W})$ . A CDAG where  $\mathbf{W}$  is empty is written as  $\mathcal{G}(\mathbf{V})$  or  $\mathcal{G}(\mathbf{O}, \mathbf{M})$  as a shorthand.

A conditional acyclic directed mixed graph (CADMG)  $\mathcal{G}(\mathbf{V} \mid \mathbf{W})$  is a mixed graph with two types of edges  $\rightarrow$  and  $\leftrightarrow$  with no directed cycles, where no arrowhead may point to an element of  $\mathbf{W}$ . We will sometimes omit variables from CDAGs and CADMGs if they are obvious to avoid notation clutter, e.g. we will write  $\mathcal{G}(\mathbf{V} \mid \mathbf{W})$  simply as  $\mathcal{G}$ . Given a CDAG  $\mathcal{G}(\mathbf{O}, \mathbf{M} \mid \mathbf{W})$ , define  $\mathcal{G}_{\mathbf{B}}(\mathcal{G})$  to be an edge subgraph obtained from  $\mathcal{G}$  by removing all arrows pointing away from  $\mathbf{B}$ .

Define a *latent projection* of  $\mathcal{G}(\mathbf{O}, \mathbf{M} \mid \mathbf{W})$  onto  $\mathbf{O} \cup \mathbf{W}$  [23] to be a CADMG  $\mathcal{G}_{(\mathbf{O})}(\mathbf{O}, \mathbf{M} \mid \mathbf{W}) \equiv \mathcal{G}^{\dagger}(\mathbf{O} \mid \mathbf{W})$  such that for any  $V_1, V_2 \in \mathbf{O} \cup \mathbf{W}$ :

- There is an edge  $V_1 \rightarrow V_2$  if and only if there is a directed path  $V_1 \rightarrow \dots \rightarrow V_2$  in  $\mathcal{G}(\mathbf{O}, \mathbf{M} \mid \mathbf{W})$  with all intermediate nodes in  $\mathbf{M}$ .
- There is an edge  $V_1 \leftrightarrow V_2$  if and only if there is a marginally d-connected path  $V_1 \leftarrow \dots \rightarrow V_2$  in  $\mathcal{G}(\mathbf{O}, \mathbf{M} \mid \mathbf{W})$  with all intermediate nodes in  $\mathbf{M}$ .

Latent projections are a simplified representation of an infinitely large class of hidden variable CDAGs with structural features in common. In this paper, we use them only to simplify the statements and proofs of our results. The results themselves will always be about models represented by DAGs (and CDAGs).

Given a CDAG  $\mathcal{G}(\mathbf{V} \mid \mathbf{W})$ , and  $\mathbf{A} \subseteq \mathbf{V} \cup \mathbf{W}$ , define  $\mathcal{G}_{\mathbf{A}}(\mathbf{V} \mid \mathbf{W}) \equiv \mathcal{G}(\mathbf{V} \cap \mathbf{A} \mid \mathbf{W} \cap \mathbf{A})$  be a subgraph of  $\mathcal{G}$  containing the vertex set  $\mathbf{A}$  and any edge in  $\mathcal{G}$  between elements in  $\mathbf{A}$ .

Given a CADMG  $\mathcal{G}(\mathbf{V} \mid \mathbf{W})$ , and  $V \in \mathbf{V}$ , define the *district* (or *c-component* [22, 18]) of  $V$  in  $\mathcal{G}(\mathbf{V} \mid \mathbf{W})$  to be  $\text{dis}_{\mathcal{G}}(V) = \{A \in \mathbf{V} \mid V \leftrightarrow \dots \leftrightarrow A\}$ . The set of districts of  $\mathcal{G}(\mathbf{V} \mid \mathbf{W})$  is denoted by  $\mathcal{D}(\mathcal{G}(\mathbf{V} \mid \mathbf{W}))$ , and it partitions  $\mathbf{V}$ .

For any  $V \in \mathbf{O}$  in a CDAG  $\mathcal{G}(\mathbf{O}, \mathbf{M} \mid \mathbf{W})$  where for every  $M \in \mathbf{M}$ ,  $\text{de}_{\mathcal{G}}(M) \cap \mathbf{O} \neq \emptyset$ , define the *clan* of  $V$  as  $\text{cl}_{\mathcal{G}}(V) \equiv \text{an}_{\mathcal{G}_{\mathbf{D}_{\mathbf{V} \cup \mathbf{M}}}}(\mathbf{D}_V)$ , where  $\mathbf{D}_V = \text{dis}_{\mathcal{G}_{(\mathbf{O})}}(V)$ . For example, in  $\mathcal{G}$  shown in Fig. 1 (c), where  $\{X, W\}$  are missing,  $\text{cl}_{\mathcal{G}}(R_X) = \text{cl}_{\mathcal{G}}(S_W) = \{W, R_X, S_W\}$ , and  $\text{cl}_{\mathcal{G}}(R_W) = \text{cl}_{\mathcal{G}}(S_X) = \{X, R_W, S_X\}$ .

For any  $\mathbf{D} \in \mathcal{D}(\mathcal{G}_{(\mathbf{O})}(\mathbf{O}, \mathbf{M} \mid \mathbf{W}))$ , and  $D_1, D_2 \in \mathbf{D}$ ,  $\text{cl}_{\mathcal{G}}(D_1) = \text{cl}_{\mathcal{G}}(D_2)$ . Thus we will write  $\text{cl}_{\mathcal{G}}(\mathbf{D}) \equiv \text{cl}_{\mathcal{G}}(D)$ , for any  $D \in \mathbf{D}$ . In fact, the set of clans partitions  $\mathbf{O} \cup \mathbf{M}$  in  $\mathcal{G}$  with the property above.

Given a CDAG  $\mathcal{G}$ , a total ordering  $\prec$  on vertices in  $\mathcal{G}$  is *topological given  $\mathcal{G}$*  if  $A \prec B$  implies  $A \notin \text{de}_{\mathcal{G}}(B)$ . Given an ordering  $\prec$  topological given  $\mathcal{G}$ , define for any vertex  $V$

in  $\mathcal{G}$ ,  $\text{pre}_{\mathcal{G}, \prec}(V) = \{W \neq V \mid W \prec V\}$ . Given  $\prec$  topological for  $\mathcal{G}$  with a vertex set  $\mathbf{V}$ , if there is a subgraph  $\mathcal{G}'$  of  $\mathcal{G}$  with a vertex set  $\mathbf{V}' \subset \mathbf{V}$ , we will view  $\prec$  with respect to  $\mathcal{G}'$  as the natural subordering restricted to  $\mathbf{V}'$ . Note that this subordering will also be topological with respect to  $\mathcal{G}'$ .

A counterfactual (potential outcome)  $Y(\mathbf{a})$  [8, 14] is a response  $Y$  to a hypothetical assignment of a set of treatments  $\mathbf{A}$  to values  $\mathbf{a}$ . Given a set of potential outcomes  $Y_1(\mathbf{a}), \dots, Y_k(\mathbf{a})$ , where  $\mathbf{Y} = \{Y_1, \dots, Y_k\}$ , we may consider a joint distribution

$$p(\{Y_1, \dots, Y_k\}(\mathbf{a})) \equiv p(\mathbf{Y}(\mathbf{a})) \equiv p(\mathbf{Y} \mid \text{do}(\mathbf{a})).$$

The  $\text{do}(\cdot)$  notation is discussed extensively in [10].

### 3 MISSING GRAPHS AND MISSINGNESS MODELS

Given a CDAG  $\mathcal{G}(\mathbf{V} \mid \mathbf{W})$ , we say  $p_{\mathbf{W}}(\mathbf{V})$  (a mapping from  $\mathfrak{X}_{\mathbf{W}}$  to  $p(\mathbf{V})$ ) is Markov relative to  $\mathcal{G}$  if

$$p_{\mathbf{W}}(\mathbf{V}) = \prod_{V \in \mathbf{V}} p_{\mathbf{W}}(V \mid \text{pa}_{\mathcal{G}}(V) \setminus \mathbf{W}), \quad (1)$$

and each term  $p_{\mathbf{W}}(V \mid \text{pa}_{\mathcal{G}}(V) \setminus \mathbf{W})$  only depends on  $\mathbf{W} \cap \text{pa}_{\mathcal{G}}(V)$ .

**Definition 1 (missingness graph)** *Given a DAG  $\mathcal{G}(\mathbf{O}, \mathbf{M})$ , a DAG  $\mathcal{G}^m$  is called a missingness graph for  $\mathcal{G}$  if  $\mathcal{G}^m$  has the vertex set  $\mathbf{O} \cup \mathbf{M} \cup \mathbf{R}_{\mathbf{M}} \cup \mathbf{S}_{\mathbf{M}}$ , where  $\mathbf{R}_{\mathbf{M}} = \{R_M \mid M \in \mathbf{M}\}$ ,  $\mathbf{S}_{\mathbf{M}} = \{S_M \mid M \in \mathbf{M}\}$ ,  $\mathcal{G} = \mathcal{G}_{\mathbf{O} \cup \mathbf{M}}^m$ , and for all  $M$  in  $\mathbf{M}$ ,  $\text{pa}_{\mathcal{G}^m}(S_M) = \{M, R_M\}$ ,  $\text{ch}_{\mathcal{G}^m}(S_M) = \emptyset$ , and  $\text{ch}_{\mathcal{G}^m}(R_M) \cap (\mathbf{O} \cup \mathbf{M}) = \emptyset$ .*

By convention, if  $\mathbf{M} = \emptyset$ , then  $\mathbf{S}_{\emptyset} = \mathbf{R}_{\emptyset} = \emptyset$ . We will refer to  $\mathbf{O} \cup \mathbf{R}_{\mathbf{M}} \cup \mathbf{S}_{\mathbf{M}}$  as  $\mathbf{V}$ , and to  $\mathbf{V} \cup \mathbf{M}$  as  $\mathbf{A}$ . We call elements of  $\mathbf{R}_{\mathbf{M}}$  indicators, and elements of  $\mathbf{S}_{\mathbf{M}}$  proxies.

Define  $\mathcal{M}(\mathcal{G}^m(\mathbf{A}))$  to be the *missingness model* for a missingness graph  $\mathcal{G}^m(\mathbf{A})$  as a set of distributions  $\{p(\mathbb{A})\}$  over the following set of counterfactuals  $\mathbb{A} \equiv \{\mathbf{A}(\mathbf{r}) \mid \mathbf{R} \subseteq \mathbf{R}_{\mathbf{M}}, \mathbf{r} \in \mathfrak{X}_{\mathbf{R}}\}$ , such that  $(\forall M \in \mathbf{M}) \mathfrak{X}_{R_M} = \{0, 1\}$ ,  $\mathfrak{X}_{S_M} = \mathfrak{X}_M \cup \{\text{missing}\}$ , and the missingness mechanism that determines the value of  $S_M$  is as follows:  $S_M(0_{R_M}) = M$  and  $S_M(1_{R_M}) = \text{missing}$ . In addition:  $(\forall \mathbf{R} \subseteq \mathbf{R}_{\mathbf{M}}, \mathbf{r} \in \mathfrak{X}_{\mathbf{R}}, V \in \mathbf{A})$ ,

$$V(\mathbf{r}) \perp\!\!\!\perp \{\text{ndp}_{\mathcal{G}^m_{\mathbf{R}}}(V)\}(\mathbf{r}) \mid \{\text{pa}_{\mathcal{G}^m_{\mathbf{R}}}(V)\}(\mathbf{r}). \quad (2)$$

To obtain the set  $\mathbb{A}$ , we first define

$$\{\mathbf{A}(\mathbf{r}) \mid \mathbf{r} \in \mathfrak{X}_{\mathbf{R}_{\mathbf{M}}}\} \equiv \{\mathbf{S}_{\mathbf{M}}(\mathbf{r}), \mathbf{O}, \mathbf{M} \mid \mathbf{r} \in \mathfrak{X}_{\mathbf{R}_{\mathbf{M}}}\},$$

and obtain the others via modified recursive substitution as in definition 43 in [11], pp. 100-101.

A missingness model is thus really a particular type of a graphical causal model where we only define interventions on a subset of variables [11]. In particular, we allow

$\mathcal{G}(\mathbf{O}, \mathbf{M})$  to represent an ordinary hidden variable statistical model. (2) is just the DAG local Markov property linking  $p(\mathbf{A}(\mathbf{r}))$  and  $\mathcal{G}_{\mathbf{R}}^m$ , for every  $\mathbf{r}$ . If we had chosen to split variables in  $\mathbf{R}$  into random and intervened versions, and display both explicitly in the graph rather than only displaying the random version of variables, and keeping intervened versions implicit, as we do in  $\mathcal{G}_{\mathbf{R}}^m$ , we would end up with Single World Intervention Graphs (SWIGs), and the appropriate local Markov property for those graphs, as discussed in [11].

Standard results on DAG models imply (2) is equivalent to (1) for  $p(\mathbf{A}(\mathbf{r}))$  and  $\mathcal{G}_{\mathbf{R}}^m$  (if we let  $\mathbf{W} = \emptyset$ , and keep fixed versions of  $\mathbf{R}$  implicit in the graph). We may also let  $\mathbf{W} = \mathbf{R}$ , and treat  $\mathbf{R}$  as a split node as in a SWIG.

### 4 RECOVERABILITY

We call  $p(\mathbf{V})$  the *manifest distribution*. A functional of  $p(\mathbb{A})$ ,  $f(p(\mathbb{A}))$  is said to be *recoverable* given  $p(\mathbf{V})$  in  $\mathcal{G}^m$  if there is a functional  $g$  of  $p(\mathbf{V})$ , such that  $f(p(\mathbb{A})) = g(p(\mathbf{V}))$  for every element of  $\mathcal{M}(\mathcal{G}^m)$ . In this paper, we will concentrate on recoverability of  $p(\mathbf{O} \cup \mathbf{M})$ , although many other kinds of recoverability problems are also interesting, for instance recovering the causal effect in a causal model with missingness.

We explicitly represent missingness as a causal inference problem because this allows us to rephrase recoverability as identifiability of causal effects. If we were allowed to assign  $\mathbf{R}_{\mathbf{M}}$  without affecting other variables, we could use proxies  $\mathbf{S}_{\mathbf{M}}$  to recover the behavior of the underlying missing variables  $\mathbf{M}$ , due to the following result.

**Lemma 1** *In a DAG  $\mathcal{G}$  where  $\mathbf{M} \neq \emptyset$ , for any  $p(\mathbb{A}) \in \mathcal{M}(\mathcal{G}^m(\mathbf{V}, \mathbf{M}))$ , and  $R_M \in \mathbf{R}_{\mathbf{M}}$ ,  $p(\mathbb{Y}) \in \mathcal{M}(\mathcal{G}^m(\mathbf{V} \cup \{M\}, \mathbf{M} \setminus \{M\})_{\mathbf{V} \cup \mathbf{M} \setminus \{S_M, R_M\}})$ , where  $\mathbb{Y}$  is*

$$\{\{\mathbf{V} \cup \mathbf{M} \setminus \{R_M\}\}(\mathbf{r}, 0_{R_M}) \mid \mathbf{R} \subseteq \mathbf{R}_{\mathbf{M} \setminus \{M\}}, \mathbf{r} \in \mathfrak{X}_{\mathbf{R}}\}.$$

*Proof:*  $\{\mathbf{V} \cup \mathbf{M}\}(\mathbf{r}, 0_{R_M})$  obeys (2) for  $\mathcal{G}_{\mathbf{R} \cup \{R_M\}}^m$ . Since  $\mathbf{A} \setminus \{R_M\}$  is ancestral in  $\mathcal{G}_{\mathbf{R} \cup \{R_M\}}^m$ ,  $\{\mathbf{V} \cup \mathbf{M} \setminus \{R_M\}\}(\mathbf{r}, 0_{R_M})$  obeys (2) for  $(\mathcal{G}_{\mathbf{R} \cup \{R_M\}}^m)_{\mathbf{A} \setminus \{R_M\}}$ . Our conclusion follows since  $M = S_M(0_{R_M})$ .  $\square$

In other words, fixing  $R_M$  to 0 gives a new model where  $M$  is effectively observed since  $M = S_M(0_{R_M})$ . This implies that if we were able to fix all of  $\mathbf{R}_{\mathbf{M}}$ , we could recover  $p(\mathbf{O} \cup \mathbf{M})$ .

**Corollary 1**  $p(\{\mathbf{O}, \mathbf{S}_{\mathbf{M}}\}(\mathbf{0}_{\mathbf{R}_{\mathbf{M}}})) = p(\mathbf{O} \cup \mathbf{M})$  for any  $\mathcal{G}^m$ , and any  $p(\mathbb{A}) \in \mathcal{M}(\mathcal{G}^m)$ .

This corollary implies that our recoverability problem is solved by expressing a particular interventional distribution

as a function of the manifest in a restricted causal model. We will attack this problem via two standard results for causal models that hold in restricted causal models as well, as shown in [11], propositions 45 and 46.

**Theorem 1** For any  $p(\mathbb{A}) \in \mathcal{M}(\mathcal{G}^m(\mathbf{V}, \mathbf{M}))$ , and  $(\forall \mathbf{R} \subseteq \mathbf{R}_M, \mathbf{r} \in \mathfrak{X}_R)$ ,

$$p(\mathbf{A}(\mathbf{r})) = \prod_{V \in \mathbf{A}} p(V \mid \text{pa}_{\mathcal{G}^m}(V) \setminus \mathbf{R}, \mathbf{r}_{\text{pa}_{\mathcal{G}^m}(V) \cap \mathbf{R}}). \quad (3)$$

**Theorem 2** For any  $p(\mathbb{A}) \in \mathcal{M}(\mathcal{G}^m(\mathbf{V}, \mathbf{M}))$ , and  $(\forall \mathbf{R} \subseteq \mathbf{R}_M, \mathbf{r} \in \mathfrak{X}_R)$ ,

$$p(\{(\mathbf{V} \cup \mathbf{M}) \setminus \mathbf{R}\}(\mathbf{r}) \mid \mathbf{r}) = p((\mathbf{V} \cup \mathbf{M}) \setminus \mathbf{R} \mid \mathbf{r}). \quad (4)$$

(3) is known as the truncated factorization [10], manipulated distribution [21], or the g-formula [12]. (4) is known as the *consistency* property.

We now illustrate how constraints of the missingness model encoded by  $\mathcal{G}^m$ , as well as (3) and (4) lead to recoverability.

#### 4.1 EXAMPLES OF RECOVERABILITY

Consider Fig. 1, where  $X, C, W$  may possibly be high-dimensional. In Fig. 1 (a),  $X$  is missing according to a mechanism governed by an independent proxy  $R_X$ , so

$$p(X) = p(S_X(0_{R_X})) = p(S_X \mid R_X = 0).$$

The assumption present in this model which allows us to recover the underlying missing variable, namely  $(S_X(0_{R_X}) \perp\!\!\!\perp R_X)$  is known as *missing completely at random* (MCAR) assumption.<sup>1</sup> This assumption is the missingness analogue of *ignorability* (lack of confounding between the missingness indicator  $R_X$  and the proxy  $S_X(r)$  under assignment  $r$  to  $R_X$ ).

In Fig. 1 (b),  $X$  is missing according to a mechanism governed by a proxy  $R_X$  which has a (statistical) dependence on  $X$  through  $C$ , which is a fully observed variable. In this case,

$$p(X, C) = p(S_X(0_{R_X}) \mid C)p(C) = p(S_X \mid R_X = 0, C)p(C).$$

The assumption present in this model which allows us to recover the underlying missing variable, namely  $(S_X(0_{R_X}) \perp\!\!\!\perp R_X \mid C)$  is known as the *missing at random* (MAR) assumption. This assumption is the missingness analogue of *conditional ignorability* (lack of confounding between the indicator  $R_X$  and the proxy  $S_X(r)$  under assignment  $r$  to  $R_X$  given that we conditioned on a set of variables  $C$ ).

In Fig. 1 (c), it is not the case that

$$\{S_W(0_{R_W}), S_X(0_{R_X})\} \perp\!\!\!\perp \{R_X, R_W\}.$$

<sup>1</sup>  $\perp\!\!\!\perp$  is the independence symbol.

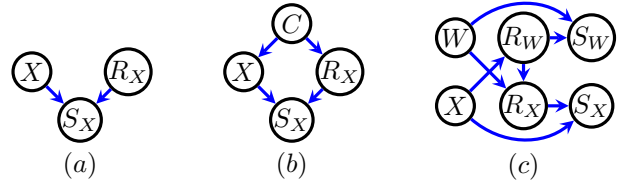


Figure 1: (a) A missingness model satisfying the *missing completely at random* (MCAR) assumption. (b) A missingness model satisfying the *missing at random* (MAR) assumption. (c) A missingness model where missingness is *not at random* (MNAR), but where recoverability is nevertheless possible.

That is, data on  $X, W$  is not missing completely at random (nor at random, since there is no fully observed variable to screen off the dependence of proxies under indicator assignment from indicators.) Nevertheless, despite the fact that data on  $p(X, W)$  is missing not at random (MNAR), we now show that  $p(X, W)$  is recoverable. We will exploit the fact that the missingness model implies

$$\{S_W(0_{R_W}), R_X(0_{R_X})\} \perp\!\!\!\perp \{S_X(0_{R_X}), R_W\}. \quad (5)$$

It is not difficult to show that  $p(R_W, R_X, S_W, S_X)$  is equal to

$$p(\{S_W, R_X\}(R_W)) \cdot p(S_X(R_X), R_W) = (p(S_W \mid R_X, R_W)p(R_X \mid R_W)) \cdot (p(S_X \mid R_X, R_W)p(R_W))$$

This implies  $p(X, W) = p(X)p(W)$  is equal to

$$\left( \sum_{R_X} p(\{S_W, R_X\}(0_{R_W})) \right) \cdot \left( \sum_{R_W} p(S_X(0_{R_X}), R_W) \right) = p(S_W \mid 0_{R_W}) \cdot \left( \sum_{R_W} p(S_X \mid 0_{R_X}, R_W)p(R_W) \right)$$

The key to this example is the joint independence (5); independences of this type arise in hidden variable DAG models. We give an example later where recoverability is based not on an ordinary independence, but on a generalized independence, or Verma constraint [23, 20]. In the following sections, we give a general recursive scheme for solving recoverability problems under MNAR using these types of constraints.

#### 4.2 KNOWN RESULTS FOR MISSINGNESS GRAPHS

Recently [7] and [6] have used missingness graphs to derive conditions for recoverability when data is MNAR. In particular, the following characterization appears in [7] (as theorem 2).

**Theorem 3** For any  $p(\mathbb{A}) \in \mathcal{M}(\mathcal{G}^m(\mathbf{V}, \mathbf{M}))$ , if no elements of  $\mathbf{R}_M$  are adjacent in  $\mathcal{G}^m(\mathbf{V}, \mathbf{M})$ , then  $p(\mathbf{O} \cup \mathbf{M})$  is recoverable from  $p(\mathbf{O}, \mathbf{S}_M, \mathbf{0}_{\mathbf{R}_M})$  if and only if  $M \notin \text{pa}_{\mathcal{G}}(R_M)$  for any  $M \in \mathbf{M}$ . Moreover,  $p(\mathbf{O} \cup \mathbf{M})$  is equal to

$$\frac{p(\mathbf{O}, \mathbf{S}_M, \mathbf{0}_{\mathbf{R}_M})}{\prod_{R_M \in \mathbf{R}_M} p(0_{R_M} \mid \text{pa}_{\mathcal{G}}(R_M) \setminus \mathbf{M}, \mathbf{S}_{\text{pa}_{\mathcal{G}}(R_M) \cap \mathbf{M}}, \mathbf{0}_{\text{pa}_{\mathcal{G}}(R_M) \cap \mathbf{M}})}$$

This result can be generalized in three directions. We may consider cases where variables are unobserved and no missingness mechanism exists. We may consider recoverability of other queries than  $p(\mathbf{O} \cup \mathbf{M})$ , for instance causal effects or marginal distributions. Finally, we may consider cases where elements of  $\mathbf{R}_M$  are adjacent. This case is important because it represents important classes of missingness such as monotonic missingness due to loss to followup. A unit that drops out of a longitudinal study at time  $t$  often remains dropped out at times  $t+1, \dots$ . In our framework, we would code this by requiring that for all  $t' > t$ ,  $R_{M_{t'}} = 1$  if  $R_{M_{t'-1}} = 1$ , where  $M_t$  is unit's status at time  $t$ . But this coding is only possible if indicators are allowed to be adjacent in the graph. In addition, allowing indicators to be adjacent allows us to model *non-monotone missing data*, where a unit may be missing at a particular time  $t$ , but then becomes observed at a later time  $t+k$ .

In this paper, we consider the problem of recovering  $p(\mathbf{O} \cup \mathbf{M})$  given that every missing variable has an indicator and a proxy (e.g. no completely hidden variables), and that indicators  $\mathbf{R}_M$  are allowed to be adjacent. We give a recoverability algorithm that generalizes earlier work in this setting.

## 5 A GENERAL RECOVERABILITY ALGORITHM

The algorithm, which we call **MID**, work as follows. It tries, for every  $R_M \in \mathbf{R}_M$ , to recover

$$p(0_{R_M} \mid \text{pa}_{\mathcal{G}^m}(R_M) \setminus \mathbf{R}_M, \mathbf{0}_{\text{pa}_{\mathcal{G}^m}(R_M) \cap \mathbf{R}_M})$$

via a subroutine **DIR**. If every such conditional distribution is recovered, **MID** recovers  $p(\mathbf{O} \cup \mathbf{M})$  via (3), otherwise **MID** fails.

The subroutine **DIR** (so named for its resemblance to the way the **ID** algorithm operates when identifying controlled direct effects) has three cases. The first case, which is sufficient for obtaining the soundness part of Theorem 3, attempts to check if indicators for missing parents of  $R_M$  are non-parental non-descendants of  $R_M$ , in which case recoverability of the conditional distribution for  $R_M$  is immediate.

Otherwise, **DIR** uses the other two cases to isolate  $R_M$  and its parents into smaller subproblems based on a particular type of ancestral set  $\mathbf{A}^\dagger$ , or the clan  $\mathbf{D}^\dagger$  of  $R_M$ . **DIR**

is recursive, which means the input must also keep track of a set  $\mathbf{W}$  representing variables the clan subproblem ends up depending on.

The situation is somewhat analogous to the way in which the **ID** algorithm attempts to identify controlled direct effects  $p(Y \mid \text{do}(\mathbf{v}_{\text{pa}_{\mathcal{G}}(Y)})) = p(Y(\mathbf{v}_{\text{pa}_{\mathcal{G}}(Y)}))$ , with three major differences. First, we are attempting identification in a setting where some variables start off being treated as hidden, but in the course of the recursion of **DIR** become observed due to fixing indicators to 0. In **ID** variables are always either hidden or observed and do not change status. Second, since we are only allowed to intervene on indicators, we are attempting to identify

$$p(R_M(\mathbf{0}_{\mathbf{R}_M \cap \text{pa}_{\mathcal{G}}(R_M)} \mid \{\text{pa}_{\mathcal{G}}(R_M) \setminus \mathbf{R}_M\}(\mathbf{0}_{\mathbf{R}_M \cap \text{pa}_{\mathcal{G}}(R_M)}))).$$

Finally, there is not necessarily a fully interventional interpretation for the intermediate objects  $p_{\mathbf{W}}(\cdot)$  that arise during the execution of **DIR**, since  $\mathbf{W}$  may contain elements outside  $\mathbf{R}_M$ . This is a necessary consequence of our insistence on not imposing a causal model on  $p(\mathbf{M} \cup \mathbf{O})$ . Intermediate objects that arise during the execution of **ID** can always be interpreted as interventional distributions.

### 5.1 SOUNDNESS

**MID** and its subroutine **DIR** appear below as algorithm 1. In this section, we prove that **MID** is sound.

Corollary 1 implies that if we were able to express  $p(\{\mathbf{O}, \mathbf{S}_M\}(\mathbf{0}_{\mathbf{R}_M}))$  as a function of the manifest distribution, we would solve the recoverability problem for  $p(\mathbf{O} \cup \mathbf{M})$ . If we happen to know

$$p(0_{R_M} \mid \text{pa}_{\mathcal{G}^m}(R_M) \setminus \mathbf{R}_M, \mathbf{0}_{\text{pa}_{\mathcal{G}^m}(R_M) \cap \mathbf{R}_M})$$

for every  $R_M \in \mathbf{R}_M$  as a function of the manifest, this would suffice due to the following result.

**Lemma 2** Under  $\mathcal{M}(\mathcal{G}^m)$ , if for every  $R_M \in \mathbf{R}_M$ ,

$$p(0_{R_M} \mid \text{pa}_{\mathcal{G}^m}(R_M) \setminus \mathbf{R}_M, \mathbf{0}_{\text{pa}_{\mathcal{G}^m}(R_M) \cap \mathbf{R}_M})$$

is a functional  $f_{R_M}(\cdot)$  of  $p(\mathbf{O}, \mathbf{S}_M, \mathbf{0}_{\mathbf{R}_M})$ , then

$$p(\{\mathbf{O}, \mathbf{S}_M\}(\mathbf{0}_{\mathbf{R}_M})) = \frac{p(\mathbf{O}, \mathbf{S}_M, \mathbf{0}_{\mathbf{R}_M})}{\prod_{R_M \in \mathbf{R}_M} f_{R_M}(p(\mathbf{O}, \mathbf{S}_M, \mathbf{0}_{\mathbf{R}_M}))}.$$

*Proof:*  $p(\{\mathbf{O}, \mathbf{S}_M\}(\mathbf{0}_{\mathbf{R}_M})) = \sum_{\mathbf{M}} p(\{\mathbf{A} \setminus \mathbf{R}_M\}(\mathbf{0}_{\mathbf{R}_M}))$ .

$$p(\{\mathbf{A} \setminus \mathbf{R}_M\}(\mathbf{0}_{\mathbf{R}_M})) = \frac{p(\mathbf{A} \setminus \mathbf{R}_M, \mathbf{0}_{\mathbf{R}_M})}{\prod_{R_M \in \mathbf{R}_M} f_{R_M}(p(\mathbf{O}, \mathbf{S}_M, \mathbf{0}_{\mathbf{R}_M}))}$$

is implied by (3). But no denominator is a function of  $\mathbf{M}$ , so we can apply the sum to the numerator first.  $\square$  Finding functionals  $f_{R_M}(\cdot)$  for every  $R_M$  in order to apply Lemma 2 is the job of the subroutine **DIR**.



**Algorithm 1**  $\mathcal{G}^m(\mathbf{V}, \mathbf{M})$  a missingness graph,  $p(\mathbf{V})$  a manifest distribution from  $p(\mathbb{A}) \in \mathcal{M}(\mathcal{G}^m(\mathbf{V}, \mathbf{M}))$ ,  $p_{\mathbf{W}}(\mathbf{V})$  a family of manifest distributions from elements of  $p(\mathbb{A}) \in \mathcal{M}(\mathcal{G}^m(\mathbf{V}, \mathbf{M}))$ ,  $\prec$  a topological order on  $\mathcal{G}^m$ .

**procedure** MID( $\mathcal{G}^m(\mathbf{V}, \mathbf{M}), p(\mathbf{V})$ )

**for each**  $R_M \in \mathbf{R}_M$ ,

$\tilde{p}(0_{R_M} \mid \text{pa}_{\mathcal{G}^m}(R_M) \setminus \mathbf{R}_M, \mathbf{0}_{\text{pa}_{\mathcal{G}^m}(R_M) \cap \mathbf{R}_M})$   
 $\leftarrow \text{DIR}(\mathcal{G}^m, p, R_M)$

**if**  $(\exists R_M \in \mathbf{R}_M)$ , s.t.  $\text{DIR}(\mathcal{G}^m, p, R_M) = \emptyset$ ,

**return** “cannot recover.”

**else return**

$$\frac{p(\mathbf{O}, \mathbf{S}_M, \mathbf{0}_{\mathbf{R}_M})}{\prod_{R_M \in \mathbf{R}_M} \tilde{p}(0_{R_M} \mid \text{pa}_{\mathcal{G}^m}(R_M) \setminus \mathbf{R}_M, \mathbf{0}_{\text{pa}_{\mathcal{G}^m}(R_M) \cap \mathbf{R}_M})}.$$

**end procedure**

**procedure** DIR( $\mathcal{G}^m(\mathbf{V}, \mathbf{M} \mid \mathbf{W}), p_{\mathbf{W}}(\mathbf{V}), R_M$ )

**if**  $\mathbf{R}_M \cap (\text{pa}_{\mathcal{G}^m}(R_M) \setminus \mathbf{W}) \subseteq \text{ndp}_{\mathcal{G}^m}(R_M)$ , **return**

$$p_{\mathbf{W}} \left( 0_{R_M} \mid \begin{array}{l} \text{pa}_{\mathcal{G}^m}(R_M) \setminus (\mathbf{M} \cup \mathbf{W} \cup \mathbf{R}_M) \\ \mathbf{0}_{\mathbf{R}_M \cap \text{pa}_{\mathcal{G}^m}(R_M)}, \mathbf{S}_{\mathbf{M} \cap \text{pa}_{\mathcal{G}^m}(R_M)} \\ \mathbf{0}_{\mathbf{R}_M \cap (\text{pa}_{\mathcal{G}^m}(R_M) \setminus \mathbf{W})} \end{array} \right).$$

**else**  $\mathbf{A}^\dagger \leftarrow \{R_M\}$ .

**while**  $(\text{an}_{\mathcal{G}^m}(\mathbf{A}^\dagger) \cup \mathbf{S}_{\text{an}_{\mathcal{G}^m}(\mathbf{A}^\dagger) \cap \mathbf{M}} \not\subseteq \mathbf{A}^\dagger)$  **do**

$\mathbf{A}^\dagger \leftarrow \text{an}_{\mathcal{G}^m}(\mathbf{A}^\dagger) \cup \mathbf{S}_{\text{an}_{\mathcal{G}^m}(\mathbf{A}^\dagger) \cap \mathbf{M}}$ .

**if**  $\mathbf{A}^\dagger \subset \mathbf{A}$ ,

**return** DIR( $\mathcal{G}_{\mathbf{A}^\dagger}^m, p_{\mathbf{W} \cap \mathbf{A}^\dagger}(\mathbf{V} \cap \mathbf{A}^\dagger), R_M$ ).

$\mathbf{D} \leftarrow \text{dis}_{\mathcal{G}^m(\mathbf{V})}(R_M)$ ,  $\mathbf{D}^\dagger \leftarrow \text{cla}_{\mathcal{G}^m}(R_M)$ .

**if**  $\mathbf{D} \subset \mathbf{V}$ ,

$\mathbf{Z}^\dagger \leftarrow \text{pa}_{\mathcal{G}^m(\mathbf{V})}^s(\mathbf{D}) \cap \mathbf{R}_M$

$\mathbf{Y}^\dagger \leftarrow \text{pa}_{\mathcal{G}^m(\mathbf{V})}^s(\mathbf{D}) \setminus \mathbf{R}_M$

$\mathbf{M}_{\mathbf{D}^\dagger}^o \leftarrow \{M \in (\mathbf{M} \cap \mathbf{D}^\dagger) \mid R_M \in \mathbf{Z}^\dagger\}$

$\mathbf{M}_{\mathbf{D}^\dagger}^h \leftarrow (\mathbf{M} \cap \mathbf{D}^\dagger) \setminus \mathbf{M}_{\mathbf{D}^\dagger}^o$

$\mathbf{V}^\dagger \leftarrow \mathbf{D} \cup \mathbf{M}_{\mathbf{D}^\dagger}^o$

$\tilde{\mathcal{G}}^m \leftarrow \mathcal{G}_{\mathbf{D}^\dagger}^m(\mathbf{V}^\dagger, \mathbf{M}_{\mathbf{D}^\dagger}^h \mid \mathbf{Y}^\dagger)$

$$p_{\mathbf{Y}^\dagger}(\mathbf{D}) \leftarrow \prod_{V \in \mathbf{D}} p_{\mathbf{W}} \left( V \mid \begin{array}{l} \text{pre}_{\mathcal{G}^m(\mathbf{V}), \prec}(V) \setminus \mathbf{Z}^\dagger, \\ \mathbf{0}_{\text{pre}_{\mathcal{G}^m(\mathbf{V}), \prec}(V) \cap \mathbf{Z}^\dagger} \end{array} \right)$$

**return** DIR( $\tilde{\mathcal{G}}^m, p_{\mathbf{Y}^\dagger}(\mathbf{D}), R_M$ )

**end if**

**return**  $\emptyset$ .

**end procedure**

## Soundness of DIR

The subroutine **DIR** invoked by **MID** aims to recover  $f_{R_M}(p) = p(0_{R_M} \mid \text{pa}_{\mathcal{G}^m}(R_M) \setminus \mathbf{R}_M, \mathbf{0}_{\text{pa}_{\mathcal{G}^m}(R_M) \cap \mathbf{R}_M})$  by recursively attempting to restrict  $R_M$  and  $\text{pa}_{\mathcal{G}^m}(R_M)$  to either an appropriate ancestral subset containing these vertices, or an appropriate clan of  $\mathcal{G}^m$ , and, in the base case, exploiting the independence structure, and properties of the subproblem that is left.

To prove the soundness of **DIR**, we must establish, by induction on algorithm structure, certain results about the subproblems it considers. We will represent subproblems as a pair consisting of a CDAG  $\tilde{\mathcal{G}}^m$  that is a subgraph of the original graph  $\mathcal{G}^m$ , and a *conditional fragment* of the missingness model which can be viewed as a set of all interventional distributions relevant to the subproblem, which also possibly depend on variables  $\mathbf{W}$  from larger subproblems.

Given an element  $p(\mathbb{A})$  of  $\mathcal{M}(\mathcal{G}^m(\mathbf{A}))$ ,  $\mathbf{B} \subseteq \mathbf{A}$ , and  $\mathbf{W} \subseteq \mathbf{A} \setminus \mathbf{B}$ , a *conditional fragment of  $p(\mathbb{A})$  with respect to  $\mathbf{B}$  and  $\mathbf{W}$* , denoted by  $\mathcal{F}_{\mathbf{W}, \mathbf{B}}$ , is a mapping from elements  $\mathbf{w}$  in  $\mathfrak{X}_{\mathbf{W}}$  to

$$\mathcal{F}_{\mathbf{w}, \mathbf{B}} \equiv \{p_{\mathbf{w}}(\mathbf{B}(\mathbf{r})_{\mathbf{w}}) \mid \mathbf{R} \subseteq \mathbf{R}_M \cap \mathbf{B}, \mathbf{r} \in \mathfrak{X}_{\mathbf{R}}\}.$$

Note that we cannot view  $p_{\mathbf{w}}(\mathbf{B}(\mathbf{r})_{\mathbf{w}})$  as a joint response of  $\mathbf{B}$  to an intervention setting  $\mathbf{R} \cup \mathbf{W}$  to  $\mathbf{r} \cup \mathbf{w}$ , because  $\mathbf{W}$  may contain elements outside  $\mathbf{R}$  that we are not allowed to intervene on.

For each call to **DIR**, we want to show that all interventional distributions in the input fragment are Markov with respect to the appropriately modified input graph, that we have enough information in the subproblem to possibly obtain  $f_{R_M}(p)$ , and that the manifest distribution of the fragment for the current (inner) call can be obtained from the manifest distribution of the fragment for the previous (outer) call.

**Definition 2**  $\mathcal{F}_{\mathbf{W}, \mathbf{B}}$  is causal Markov relative to a CDAG  $\mathcal{G}^m(\mathbf{B} \mid \mathbf{W})$  if  $(\forall \mathbf{w} \in \mathfrak{X}_{\mathbf{W}}, p_{\mathbf{w}}(\mathbf{B}(\mathbf{r})_{\mathbf{w}}) \in \mathcal{F}_{\mathbf{w}, \mathbf{B}})$ ,  $p_{\mathbf{w}}(\mathbf{B}(\mathbf{r})_{\mathbf{w}})$  is Markov relative to  $\mathcal{G}^m(\mathbf{B} \mid \mathbf{W})_{\mathbf{R}}$ .

This definition is how we will relate fragments and corresponding subgraphs, and the following two results establish this relationship for the two recursive cases relevant for **DIR**.

**Lemma 3** For  $\mathcal{F}_{\mathbf{W}, \mathbf{A}}$  causal Markov relative to  $\mathcal{G}^m(\mathbf{A} \mid \mathbf{W})$ , let  $\mathbf{D} \in \mathcal{D}(\mathcal{G}_{\mathbf{V}}^m)$ ,  $\mathbf{D}^\dagger \equiv \text{cla}_{\mathcal{G}^m}(\mathbf{D})$ ,  $\mathbf{W}^\dagger \equiv \text{pa}_{\mathcal{G}_{\mathbf{V}}^m}^s(\mathbf{D})$ ,  $\mathbf{W}^* \equiv \mathbf{W}^\dagger \setminus \mathbf{W}$ . Then for any  $\mathbf{w}^\dagger \in \mathfrak{X}_{\mathbf{W}^\dagger}$ ,  $\mathcal{F}_{\mathbf{w}^\dagger, \mathbf{D}^\dagger} \equiv \{\tilde{p}_{\mathbf{w}^\dagger}(\mathbf{D}^\dagger(\mathbf{r})_{\mathbf{w}^\dagger}) \mid \mathbf{r} \in \mathfrak{X}_{\mathbf{R}}, \mathbf{R} \subseteq \mathbf{R}_M \cap \mathbf{D}\}$  is causal Markov relative to  $\mathcal{G}_{\text{fa}_{\mathcal{G}^m}(\mathbf{D}^\dagger)}^m(\mathbf{D}^\dagger \mid \mathbf{W}^\dagger)$ , where for any  $\mathbf{w}$  consistent with  $\mathbf{w}^\dagger$ ,  $\tilde{p}_{\mathbf{w}^\dagger}(\mathbf{D}^\dagger(\mathbf{r})_{\mathbf{w}^\dagger})$  is

$$\prod_{V \in \mathbf{D}^\dagger} p_{\mathbf{w}}(V \mid (\mathbf{r} \cup \mathbf{w}^\dagger)_{\text{pa}_{\mathcal{G}^m}(V) \cap (\mathbf{R} \cup \mathbf{W}^*)}, \text{pa}_{\mathcal{G}^m}(V) \setminus (\mathbf{R} \cup \mathbf{W}^\dagger))$$

*Proof:* For any CDAG  $\mathcal{G}(\mathbf{O}, \mathbf{M} \mid \mathbf{W})$ ,  $\text{fa}_{\mathcal{G}}(\text{cla}_{\mathcal{G}}(\mathbf{D}))$  is equal to  $\text{cla}_{\mathcal{G}}(\mathbf{D}) \cup \text{pa}_{\mathcal{G}(\mathbf{O})}^s(\mathbf{D})$  for any  $\mathbf{D} \in \mathcal{D}(\mathcal{G}(\mathbf{O}))$ . The proof is now immediate. Elements  $\tilde{p}_{\mathbf{w}^\dagger}(\mathbf{D}^\dagger(\mathbf{r})_{\mathbf{w}^\dagger})$  of each  $\mathcal{F}_{\mathbf{w}^\dagger, \mathbf{D}^\dagger}$  are Markov relative to  $(\mathcal{G}_{\text{fa}_{\mathcal{G}}(\mathbf{D}^\dagger)}^m)_{\mathbf{R}}$  by construction. The definition of  $\tilde{p}_{\mathbf{w}^\dagger}(\mathbf{D}^\dagger(\mathbf{r})_{\mathbf{w}^\dagger})$  implies it is the same object for any  $\mathbf{w}$  consistent with  $\mathbf{w}^\dagger$ .  $\square$

**Lemma 4** For  $\mathcal{F}_{\mathbf{W}, \mathbf{A}}$  causal Markov relative to  $\mathcal{G}^m(\mathbf{A} \mid \mathbf{W})$ , let  $\mathbf{V}^\dagger \subseteq \mathbf{A} \cup \mathbf{W}$  be ancestral,  $\mathbf{W}^\dagger \equiv \mathbf{W} \cap \mathbf{V}^\dagger$ ,  $\mathbf{A}^\dagger \equiv \mathbf{A} \cap \mathbf{V}^\dagger$ . Then for any  $\mathbf{w}^\dagger \in \mathfrak{X}_{\mathbf{W}^\dagger}$ ,  $\mathcal{F}_{\mathbf{w}^\dagger, \mathbf{D}^\dagger} \equiv \{\tilde{p}_{\mathbf{w}^\dagger}(\mathbf{A}^\dagger(\mathbf{r})_{\mathbf{w}^\dagger}) \mid \mathbf{r} \in \mathfrak{X}_{\mathbf{R}}, \mathbf{R} \subseteq \mathbf{R}_{\mathbf{M}} \cap \mathbf{A}^\dagger\}$  is causal Markov relative to  $\mathcal{G}_{\mathbf{A}^\dagger}^m(\mathbf{A}^\dagger \mid \mathbf{W}^\dagger)$ , where for any  $\mathbf{w}$  consistent with  $\mathbf{w}^\dagger$ ,  $\tilde{p}_{\mathbf{w}^\dagger}(\mathbf{A}^\dagger(\mathbf{r})_{\mathbf{w}^\dagger})$  is

$$\prod_{V \in \mathbf{A}^\dagger} p_{\mathbf{w}}(V \mid \mathbf{r}_{\text{pa}_{\mathcal{G}^m}(V) \cap \mathbf{R}}, \text{pa}_{\mathcal{G}^m}(V) \setminus (\mathbf{R} \cup \mathbf{W}^\dagger))$$

*Proof:* Immediate. Elements  $p_{\mathbf{w}^\dagger}(\mathbf{A}^\dagger(\mathbf{r})_{\mathbf{w}^\dagger})$  of each  $\mathcal{F}_{\mathbf{w}^\dagger, \mathbf{A}^\dagger}$  are Markov relative to  $(\mathcal{G}_{\mathbf{A}^\dagger}^m)_{\mathbf{R}}$  by construction. The definition of  $\tilde{p}_{\mathbf{w}^\dagger}(\mathbf{D}^\dagger(\mathbf{r})_{\mathbf{w}^\dagger})$  implies it is the same object for any  $\mathbf{w}$  consistent with  $\mathbf{w}^\dagger$ .  $\square$

The next two results re-express  $p(R_M \mid \text{pa}_{\mathcal{G}^m}(R_M))$  from a function of the larger fragment of the outer recursive call to a function of the smaller fragment of the inner call.

**Lemma 5** Assume  $\mathcal{F}_{\mathbf{W}, \mathbf{A}}$  is causal Markov relative to  $\mathcal{G}^m(\mathbf{A} \mid \mathbf{W})$ , and  $\mathcal{F}_{\mathbf{W}^\dagger, \mathbf{D}^\dagger}$  is defined as in Lemma 3. Then for any  $R_M \in \mathbf{D}^\dagger$ ,  $p_{\mathbf{W}}(R_M \mid \text{pa}_{\mathcal{G}^m}(R_M) \setminus \mathbf{W})$  is equal to  $p_{\mathbf{W}^\dagger}(R_M \mid \text{pa}_{\mathcal{G}^m(\mathbf{D}^\dagger)}(R_M) \setminus \mathbf{W}^\dagger)$ .

*Proof:* Since  $R_M \in \mathbf{D}^\dagger$ , this follows by Lemma 3. That is,  $p_{\mathbf{W}^\dagger}(R_M \mid \text{pa}_{\mathcal{G}^m(\mathbf{D}^\dagger)}(R_M) \setminus \mathbf{W}^\dagger)$  is equal to  $p_{\mathbf{W}}(R_M \mid \mathbf{W}^\dagger_{\text{pa}_{\mathcal{G}^m}(R_M) \cap (\mathbf{W}^\dagger \setminus \mathbf{W})}, \text{pa}_{\mathcal{G}^m}(R_M) \setminus \mathbf{W}^\dagger)$ , which is equal to  $p_{\mathbf{W}}(R_M \mid \text{pa}_{\mathcal{G}^m}(R_M) \setminus \mathbf{W})$ .  $\square$

**Lemma 6** Assume  $\mathcal{F}_{\mathbf{W}, \mathbf{A}}$  is causal Markov relative to  $\mathcal{G}^m(\mathbf{A} \mid \mathbf{W})$ , and  $\mathcal{F}_{\mathbf{W}^\dagger, \mathbf{A}^\dagger}$  is defined as in Lemma 4. Then for any  $R_M \in \mathbf{A}^\dagger$ ,  $p_{\mathbf{W}}(R_M \mid \text{pa}_{\mathcal{G}^m}(R_M) \setminus \mathbf{W})$  is equal to  $p_{\mathbf{W}^\dagger}(R_M \mid \text{pa}_{\mathcal{G}^m(\mathbf{A}^\dagger)}(R_M) \setminus \mathbf{W}^\dagger)$ .

*Proof:* Since  $R_M \in \mathbf{A}^\dagger$ , this follows by Lemma 4. That is,  $p_{\mathbf{W}^\dagger}(R_M \mid \text{pa}_{\mathcal{G}^m(\mathbf{A}^\dagger)}(R_M) \setminus \mathbf{W}^\dagger)$  is equal to  $p_{\mathbf{W}}(R_M \mid \text{pa}_{\mathcal{G}^m}(R_M) \setminus \mathbf{W})$ .  $\square$

The next two results express the analogue of the manifest distribution of the smaller fragment as a function of the manifest distribution of the larger fragment. We assume  $\mathbf{M}_{\mathbf{D}^\dagger}^o, \mathbf{M}_{\mathbf{D}^\dagger}^h, \mathbf{V}^\dagger, \mathbf{Z}^\dagger, \mathbf{Y}^\dagger$ , and  $\mathcal{G}^m$  are defined as in the district case of **DIR**. Let  $\mathbf{W}^\dagger = \mathbf{Y}^\dagger \cup \mathbf{Z}^\dagger$ , and  $\mathbf{O}^\dagger = \mathbf{D} \cap \mathbf{O}$ .

**Lemma 7** Assume  $\mathcal{F}_{\mathbf{W}, \mathbf{A}}$  is causal Markov relative to  $\mathcal{G}^m(\mathbf{A} \mid \mathbf{W})$ , and  $\mathcal{F}_{\mathbf{W}^\dagger, \mathbf{D}^\dagger}$  is defined as in Lemma 3.

Then the marginal  $p_{\mathbf{Y}^\dagger, \mathbf{O}_{\mathbf{Z}^\dagger}}(\mathbf{O}^\dagger, \mathbf{M}_{\mathbf{D}^\dagger}^o, \mathbf{S}_{\mathbf{M}_{\mathbf{D}^\dagger}^h}, \mathbf{R}_{\mathbf{M}_{\mathbf{D}^\dagger}^h})$  of  $p_{\mathbf{W}^\dagger}(\mathbf{D}^\dagger) \in \mathcal{F}_{\mathbf{W}^\dagger, \mathbf{D}^\dagger}$  is equal to  $\prod_{V \in \mathbf{V}^\dagger} p_{\mathbf{W}}(V \mid \text{pre}_{\mathcal{G}^m(\mathbf{V}), \prec}(V) \setminus \mathbf{Z}^\dagger, \mathbf{0}_{\text{pre}_{\mathcal{G}^m(\mathbf{V}), \prec}(V) \cap \mathbf{Z}^\dagger})$ .

*Proof:* Fix  $\mathbf{w}$  and  $\mathbf{w}^\dagger$  consistent with  $\mathbf{w}$ , such that  $\mathbf{w}_{\mathbf{Z}^\dagger}^\dagger = \mathbf{0}$ . We get the following set of equalities, where the first is by assumption on missingness models, the second by (4), (3) and the definition of  $\mathbf{M}_{\mathbf{D}^\dagger}^o$ , the third by definition, the fourth by Lemma 3, and the last by standard results on district factorization of hidden variable DAG models found in [22]. If we range over all possible  $\mathbf{w}_{\mathbf{Y}^\dagger}^\dagger$ , the last expression reduces to  $\prod_{V \in \mathbf{V}^\dagger} p_{\mathbf{W}}(V \mid \text{pre}_{\mathcal{G}^m(\mathbf{V}), \prec}(V) \setminus \mathbf{Z}^\dagger, \mathbf{0}_{\text{pre}_{\mathcal{G}^m(\mathbf{V}), \prec}(V) \cap \mathbf{Z}^\dagger})$ .

$$\begin{aligned} & p_{\mathbf{w}^\dagger}(\mathbf{O}^\dagger, \mathbf{M}_{\mathbf{D}^\dagger}^o, \mathbf{S}_{\mathbf{M}_{\mathbf{D}^\dagger}^h}, \mathbf{R}_{\mathbf{M}_{\mathbf{D}^\dagger}^h}) \\ &= p_{\mathbf{w}^\dagger}(\mathbf{O}^\dagger, \mathbf{S}_{\mathbf{M}_{\mathbf{D}^\dagger}^o}(\mathbf{R}_{\mathbf{M}_{\mathbf{D}^\dagger}^o} = \mathbf{0}), \mathbf{S}_{\mathbf{M}_{\mathbf{D}^\dagger}^h}, \mathbf{R}_{\mathbf{M}_{\mathbf{D}^\dagger}^h}) \\ &= p_{\mathbf{w}^\dagger}(\mathbf{O}^\dagger, \mathbf{S}_{\mathbf{M}_{\mathbf{D}^\dagger}^o}, \mathbf{S}_{\mathbf{M}_{\mathbf{D}^\dagger}^h}, \mathbf{R}_{\mathbf{M}_{\mathbf{D}^\dagger}^h}) \\ &= \sum_{\mathbf{M}_{\mathbf{D}^\dagger}^h} p_{\mathbf{w}^\dagger}(\mathbf{O}^\dagger, \mathbf{S}_{\mathbf{M}_{\mathbf{D}^\dagger}^o}, \mathbf{S}_{\mathbf{M}_{\mathbf{D}^\dagger}^h}, \mathbf{R}_{\mathbf{M}_{\mathbf{D}^\dagger}^h}, \mathbf{M}_{\mathbf{D}^\dagger}^h) \\ &= \sum_{\mathbf{M}_{\mathbf{D}^\dagger}^h} \prod_{V \in \mathbf{D}^\dagger} p_{\mathbf{w}}(V \mid \mathbf{w}_{\text{pa}_{\mathcal{G}^m}(V) \cap (\mathbf{W}^\dagger \setminus \mathbf{W})}^\dagger, \text{pa}_{\mathcal{G}^m}(V) \setminus \mathbf{W}^\dagger) \\ &= \prod_{V \in \mathbf{D}} p_{\mathbf{w}}(V \mid \text{pre}_{\mathcal{G}^m(\mathbf{V}), \prec}(V) \setminus \mathbf{W}^\dagger, \mathbf{w}_{\text{pre}_{\mathcal{G}^m(\mathbf{V}), \prec}(V) \cap \mathbf{W}^\dagger}^\dagger) \end{aligned}$$

$\square$

**Lemma 8** Assume  $\mathcal{F}_{\mathbf{W}, \mathbf{A}}$  is causal Markov relative to  $\mathcal{G}^m(\mathbf{A} \mid \mathbf{W})$ , and  $\mathcal{F}_{\mathbf{W}^\dagger, \mathbf{A}^\dagger}$  is defined as in Lemma 4. Then the element  $p_{\mathbf{W}^\dagger}(\mathbf{V} \cap \mathbf{A}^\dagger)$  of  $\mathcal{F}_{\mathbf{W}^\dagger, \mathbf{A}^\dagger}$  is equal to  $\sum_{\mathbf{V} \setminus \mathbf{A}^\dagger} p_{\mathbf{W}}(\mathbf{V})$ .

*Proof:*  $p_{\mathbf{W}^\dagger}(\mathbf{V} \cap \mathbf{A}^\dagger)$  is equal to  $\sum_{\mathbf{A}^\dagger \setminus \mathbf{V}} p_{\mathbf{W}^\dagger}(\mathbf{A}^\dagger)$  (by definition), which is equal to  $\sum_{\mathbf{A}^\dagger \setminus \mathbf{V}} \sum_{\mathbf{A} \setminus \mathbf{A}^\dagger} p_{\mathbf{W}}(\mathbf{A})$  by Lemma 4. But since both  $\mathbf{V}, \mathbf{A}^\dagger$  are subsets of  $\mathbf{A}$ , this is just  $\sum_{\mathbf{A} \setminus (\mathbf{V} \cap \mathbf{A}^\dagger)} p_{\mathbf{W}}(\mathbf{A})$ , which is equal to  $\sum_{\mathbf{V} \setminus \mathbf{A}^\dagger} \sum_{\mathbf{A} \setminus \mathbf{V}} p_{\mathbf{W}}(\mathbf{A}) = \sum_{\mathbf{V} \setminus \mathbf{A}^\dagger} p_{\mathbf{W}}(\mathbf{V})$ .  $\square$

The following result establishes the validity of the base case of **DIR**, where  $p_{\mathbf{W}}(R_M \mid \text{pa}_{\mathcal{G}^m}(R_M) \setminus \mathbf{W})$  is expressed in terms of the manifest distribution for the current fragment.

**Lemma 9** Assume  $\mathcal{F}_{\mathbf{W}, \mathbf{A}}$  is causal Markov relative to  $\mathcal{G}^m(\mathbf{A} \mid \mathbf{W})$ . Then if  $\mathbf{R}_{\mathbf{M} \cap (\text{pa}_{\mathcal{G}^m}(R_M) \setminus \mathbf{W})} \subseteq \text{ndp}_{\mathcal{G}^m}(R_M)$ , then

$$p_{\mathbf{W}}(0_{R_M} \mid \text{pa}_{\mathcal{G}^m}(R_M) \setminus (\mathbf{W} \cup \mathbf{R}_{\mathbf{M}}), \mathbf{0}_{\mathbf{R}_{\mathbf{M}} \cap (\text{pa}_{\mathcal{G}^m}(R_M) \setminus \mathbf{W})})$$

is equal to  $p_{\mathbf{W}} \left( \begin{array}{c} \text{pa}_{\mathcal{G}^m}(R_M) \setminus (\mathbf{M} \cup \mathbf{W} \cup \mathbf{R}_{\mathbf{M}}) \\ 0_{R_M} \mid \mathbf{0}_{\mathbf{R}_{\mathbf{M}} \cap \text{pa}_{\mathcal{G}^m}(R_M)}, \mathbf{S}_{\mathbf{M} \cap \text{pa}_{\mathcal{G}^m}(R_M)} \\ \mathbf{0}_{\mathbf{R}_{\mathbf{M}} \cap (\text{pa}_{\mathcal{G}^m}(R_M) \setminus \mathbf{W})} \end{array} \right)$ .

*Proof:* We get the following set of equalities, where the first follows by assumption, and the fact that  $p_{\mathbf{W}}(\mathbf{A})$  is Markov relative to  $\mathcal{G}^m$ , the second is by the properties of the missingness model, and the third is by (4):

$$\begin{aligned} p_{\mathbf{W}} \left( 0_{R_M} \middle| \begin{array}{l} \text{pa}_{\mathcal{G}^m}(R_M) \setminus (\mathbf{W} \cup \mathbf{R}_M) \\ \mathbf{0}_{\mathbf{R}_M \cap (\text{pa}_{\mathcal{G}^m}(R_M) \setminus \mathbf{W})} \end{array} \right) &= \\ p_{\mathbf{W}} \left( 0_{R_M} \middle| \begin{array}{l} \text{pa}_{\mathcal{G}^m}(R_M) \setminus (\mathbf{M} \cup \mathbf{W} \cup \mathbf{R}_M) \\ \text{pa}_{\mathcal{G}^m}(R_M) \cap \mathbf{M}, \mathbf{0}_{\mathbf{R}_M \cap \text{pa}_{\mathcal{G}^m}(R_M)} \\ \mathbf{0}_{\mathbf{R}_M \cap (\text{pa}_{\mathcal{G}^m}(R_M) \setminus \mathbf{W})} \end{array} \right) &= \\ p_{\mathbf{W}} \left( 0_{R_M} \middle| \begin{array}{l} \text{pa}_{\mathcal{G}^m}(R_M) \setminus (\mathbf{M} \cup \mathbf{W} \cup \mathbf{R}_M), \\ \mathbf{S}_{\mathbf{M} \cap \text{pa}_{\mathcal{G}^m}(R_M)}, \mathbf{0}_{\mathbf{R}_M \cap \text{pa}_{\mathcal{G}^m}(R_M)} \\ \mathbf{0}_{\mathbf{R}_M \cap \text{pa}_{\mathcal{G}^m}(R_M)}, \mathbf{0}_{\mathbf{R}_M \cap (\text{pa}_{\mathcal{G}^m}(R_M) \setminus \mathbf{W})} \end{array} \right) &= \\ p_{\mathbf{W}} \left( 0_{R_M} \middle| \begin{array}{l} \text{pa}_{\mathcal{G}^m}(R_M) \setminus (\mathbf{M} \cup \mathbf{W} \cup \mathbf{R}_M) \\ \mathbf{0}_{\mathbf{R}_M \cap \text{pa}_{\mathcal{G}^m}(R_M)}, \mathbf{S}_{\mathbf{M} \cap \text{pa}_{\mathcal{G}^m}(R_M)} \\ \mathbf{0}_{\mathbf{R}_M \cap (\text{pa}_{\mathcal{G}^m}(R_M) \setminus \mathbf{W})} \end{array} \right). \end{aligned}$$

□

Before putting all these results together to show soundness of **DIR**, we must prove one additional utility lemma that shows the set  $\mathbf{A}^\dagger$  constructed by **DIR** is ancestral.

Define an automorphism from vertex sets in  $\mathcal{G}^m$ ,  $\rho_{M, \mathcal{G}^m}(\mathbf{B})$ , as  $\text{an}_{\mathcal{G}^m}(\{R_M\} \cup \mathbf{B}) \cup \mathbf{S}_{\text{an}_{\mathcal{G}^m}(\{R_M\} \cup \mathbf{B})}$ . Let  $\mathbf{A}^\dagger$  be the fixed point of  $\rho_{M, \mathcal{G}^m}$  with the starting input of the empty set.

**Lemma 10**  $\mathbf{A}^\dagger$  is an ancestral set in  $\mathcal{G}^m$ .

*Proof:* A simple proof by contradiction follows by definition of  $\rho_{M, \mathcal{G}^m}$ . □

We now show the main result of this paper.

**Theorem 4** **MID** is sound.

*Proof:* Assuming **DIR** returns the answer for every  $R_M \in \mathbf{R}_M$ , Corollary 1, and Lemma 2 ensure that **MID** recovers  $p(\mathbf{M} \cup \mathbf{O})$  from  $p(\mathbf{V})$ .

The soundness of **DIR** follows by induction on the recursive call structure. The inductive hypothesis is that the input conditional fragment  $\mathcal{F}_{\widetilde{\mathbf{W}}, \widetilde{\mathbf{A}}}$  is causal Markov relative to the appropriate graph derived from the input graph  $\widetilde{\mathcal{G}}^m$ , that the input manifest  $\widetilde{p}_{\widetilde{\mathbf{W}}}(\mathbf{V})$  is the function of the original manifest  $p(\mathbf{V})$ , and that  $\widetilde{p}_{\widetilde{\mathbf{W}}}(R_M | \text{pa}_{\widetilde{\mathcal{G}}^m}(R_M) \setminus \widetilde{\mathbf{W}}) = p(R_M | \text{pa}_{\mathcal{G}^m}(R_M))$ .

The base case trivially holds for the original inputs to **DIR**. If the inductive hypothesis is true, and **DIR** returns after the first conditional, soundness follows by Lemma 9.

If **DIR** returns after the second conditional, then Lemma 10 ensures the constructed set  $\mathbf{A}^\dagger$  is ancestral, and the induction for the following recursive call is maintained via Lemmas 4, 8 and 6.

If **DIR** returns after the third conditional, Lemma 3 ensures  $\mathcal{F}_{\widetilde{\mathbf{W}}, \mathbf{D}^\dagger}$  is causal Markov relative to  $\mathcal{G}_{\text{fa}_{\mathcal{G}^m}(\mathbf{D}^\dagger)}^m$

for all values of  $\widetilde{\mathbf{w}}$ , including those that set  $\mathbf{Z}^\dagger$  to  $\mathbf{0}$ . Lemma 5, and the inductive hypothesis ensures  $p_{\mathbf{W}^\dagger}(R_M | \text{pa}_{\mathcal{G}_{\text{fa}_{\mathcal{G}^m}(\mathbf{D}^\dagger)}}^m(R_M) \setminus \mathbf{W}^\dagger)$  is equal to  $p_{\mathbf{W}}(R_M | \text{pa}_{\mathcal{G}^m}(R_M) \setminus \mathbf{W})$ . Finally, Lemma 7 ensures the manifest for the recursive call is a function of the input manifest. In fact, because we set  $\mathbf{Z}^\dagger$  to  $\mathbf{0}$ , properties of missingness models ensure we can treat  $\mathbf{M}_{\mathbf{D}^\dagger}^o$  as observed in subsequent recursive calls, which means we no longer need to consider  $\mathbf{S}_{\mathbf{M}_{\mathbf{D}^\dagger}^o}$ .

Since induction follows for all cases, so does our conclusion. □

## 6 A COMPLEX RECOVERABLE EXAMPLE

We now work through an example where all cases of **MID** and **DIR** are necessary. Consider the graph shown in Fig. 2 (a). Here  $C$  and  $D$  are shown in green to indicate that they are fully observed. This is a more complex version of the example in Fig. 1 (c). Unlike that case, here, there are no conditional independences that hold between proxies and indicators. However, if we were to divide by  $p(D | C)$  and sum out  $C$ , in the resulting distribution  $p_D(S_A, S_B, R_A, R_B, A, B)$ , for any fixed value  $d$  of  $D$ , we would have

$$\left( \{S_A(0_{R_A}), R_B\} \perp\!\!\!\perp \{S_B(0_{R_B}), R_A(0_{R_B})\} \right)_{p_d}$$

This is a type of Verma constraint [23] or generalized independence constraint [20].

Our goal is to recover  $p(A, B, C, D)$  given the missingness model corresponding to this graph, and in particular the above constraint. We must recover  $p(0_{R_B} | A, D)$  and  $p(0_{R_A} | 0_{R_B}, B, D)$  from  $p(R_A, R_B, S_A, S_B, C, D)$ . In either case, we note that  $D$  is not an element of  $\text{cla}_{\mathcal{G}^m}(R_A) = \text{cla}_{\mathcal{G}^m}(R_B)$ , which implies we can use the clan case of **DIR** and consider a subproblem shown in Fig. 2 (b), with the corresponding manifest  $\widetilde{p}_D(R_A, R_B, S_A, S_B, C) = p(S_A, S_B, R_A, R_B | D, C)p(C)$ . In the new subproblem (for either  $R_A$  or  $R_B$ ),  $C$  is not a part of the ancestral set  $\mathbf{A}^\dagger$  constructed by **DIR** in the ancestral case, so we consider a new subproblem shown in Fig. 2 (c), with the corresponding manifest  $\widetilde{p}_D(R_A, R_B, S_A, S_B) = \sum_c \widetilde{p}_D(S_A, S_B, R_A, R_B | D, c)\widetilde{p}_D(c)$ . This new subproblem now resembles the example in Fig. 1 (c), and is solved similarly. In particular, we recover  $p(0_{R_B} | D, A)$  as

$$\frac{\widetilde{p}_D(S_A | 0_{R_A}, 0_{R_B})\widetilde{p}_D(0_{R_B})}{\sum_{R_B} \widetilde{p}_D(S_A | 0_{R_A}, R_B)\widetilde{p}_D(R_B)}$$

and  $p(0_{R_A} | 0_{R_B}, B, D)$  as  $\widetilde{p}_D(0_{R_A} | 0_{R_B}, S_B)$ . We then obtain  $p(A, B, C, D)$  by dividing the manifest distribution for observed cases  $p(0_{R_A}, 0_{R_B}, S_A, S_B, C, D)$  by the above two probabilities.

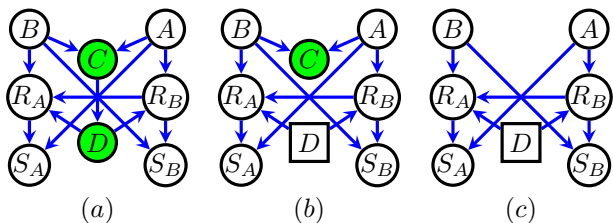


Figure 2: (a) An example where recoverability is possible via **MID**. (b),(c) Graphs corresponding to subproblems considered by **MID** in recovering  $p(A, B, C, D)$ .

## 7 NONRECOVERABILITY

The generality of **MID** naturally raises the question of whether it is complete, that is whether whenever it outputs “cannot recover” then it is possible to construct two elements of the missingness model that agree on the manifest but disagree on the underlying joint distribution. We leave this difficult question aside in this paper in the interests of space, but note that an approach similar to one used to show completeness for causal effects identification [18] seems promising. That is, use **MID** as a guide for constructing a “zoo” of structures where recoverability does not seem to be possible, and then construct a general method for showing non-recoverability for this “zoo.”

Some results on non-recoverability do exist. For example, it can be shown that  $p(A)$  is not recoverable in the missingness model with the graph in Fig. 3 (a) [7], and similarly that  $p(A, B)$  is not recoverable in the missingness model with the graph in Fig. 3 (b). Characterization of non-recoverability is an open problem.

## 8 DISCUSSION AND CONCLUSIONS

We have represented missing data as a type of a restricted causal inference problem. Using the machinery of graphical causal models, we have given a general algorithm for recoverability of a joint distribution in MNAR settings. Though we do not require this, our formalism allows the joint distribution we recover to come from a statistical, rather than a causal model – all causal assumptions may be restricted to the missingness model governing the behavior of proxies of missing variables under interventions on indicators. We show that the MCAR, MAR, MNAR taxonomy is not sufficiently granular to classify cases where recoverability is possible. In particular, there are MNAR examples where constraints akin to Verma constraints permit recoverability.

Aside from the algorithm, our formalism allows us to seamlessly integrate issues of identification of causal effects, and recoverability. For instance, it is known that in the graph shown in Fig. 3 (c) (where we treat  $\leftrightarrow$  edges as indicating

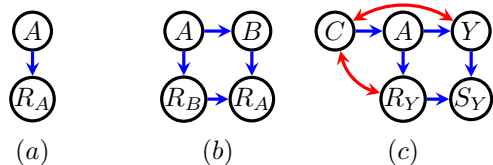


Figure 3: (a)  $p(A)$  is not recoverable. (b)  $p(A, B)$  is not recoverable. (c) A graph with hidden variables where  $p(Y)$  is not recoverable, but  $p(Y(a))$  is.

the presence of an unobserved parent),  $p(Y)$  is not recoverable. However, if the graph on  $C, A, Y$  represents a causal model, we can show that  $p(Y(a))$  is recoverable. In particular

$$p(Y(a)) = p(S_Y(a, 0_{R_Y})) = \frac{\sum_c p(S_Y, 0_{R_Y} | a, c)p(c)}{\sum_c p(0_{R_Y} | a, c)p(c)}$$

A similar observation appears in [6], example 3.

By explicitly representing missingness via an intervenable indicator, and a proxy as a response to this intervention, our formalism allows us to reason explicitly about the interpretation of censoring by death using the existing language of interventions. That is if  $S_X$  is observed patient history, and  $1_{R_X}$  implies it is missing due to the patient dying, then we may either disallow considering  $S_X(0_{R_X})$  (e.g. “resurrecting the patient”) for that patient, allow  $S_X(0_{R_X})$ , but treat it as making statements about exchangeable but different patients who happened to be alive that transfer over to the dead patient in a hypothetical alternative history where the patient never died, and so on.

Note that if we assume a *known* relationship  $p(S_M(r_{R_M}) | M)$  between  $M$  and  $S_M(r_{R_M})$  other than direct equality, we can use the approach in this paper to address certain *coarsening* [3] and *measurement error* settings. We do not consider these extensions explicitly here for space reasons, but they are straightforward.

### Acknowledgements

This research was supported in parts by grants from NIH R01 AI104459-01A1, NSF #IIS-1302448 and ONR #N00014-10-1-0933 and #N00014-13-1-0153.

### References

- [1] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39:1–38, 1977.
- [2] Constantine E. Frangakis, Donald B. Rubin, Ming-Wen An, and Ellen MacKenzie. Principal stratification designs to estimate input data missing due to death. *Biometrics*, 63:641–662, 2007.
- [3] Daniel F. Heitjan and Donald Rubin. Ignorability and coarse data. *Annals of Statistics*, 19(4):2244–2253, 1991.
- [4] Paul W. Holland. Statistics and causal inference. *Journal of the American Statistical Association*, 81:945–960, 1986.

- [5] E.L. Kaplan and P. Meier. Nonparametric estimation from incomplete observations. *Journal of the American Statistical Association*, 53:457–481, 1958.
- [6] Karthika Mohan and Judea Pearl. Graphical models for recovering probabilistic and causal queries from missing data. In Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 1520–1528. Curran Associates, Inc., 2014.
- [7] Karthika Mohan, Judea Pearl, and Jin Tian. Graphical models for inference with missing data. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 1277–1285. Curran Associates, Inc., 2013.
- [8] J. Neyman. Sur les applications de la thar des probabilités aux expériences agricoles: Essay des principe. excerpts reprinted (1990) in English. *Statistical Science*, 5:463–472, 1923.
- [9] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan and Kaufmann, San Mateo, 1988.
- [10] Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2000.
- [11] Thomas S. Richardson and Jamie M. Robins. Single world intervention graphs (SWIGs): A unification of the counterfactual and graphical approaches to causality. preprint: <http://www.csss.washington.edu/Papers/wp128.pdf>, 2013.
- [12] J.M. Robins. A new approach to causal inference in mortality studies with sustained exposure periods – application to control of the healthy worker survivor effect. *Mathematical Modeling*, 7:1393–1512, 1986.
- [13] Paul R. Rosenbaum and Donald B. Rubin. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70:41–55, 1983.
- [14] D. B. Rubin. Estimating causal effects of treatments in randomized and non-randomized studies. *Journal of Educational Psychology*, 66:688–701, 1974.
- [15] D. B. Rubin. Inference and missing data (with discussion). *Biometrika*, 63:581–592, 1976.
- [16] D. B. Rubin. *Multiple Imputation for Nonresponse in Surveys*. New York: Wiley & Sons, 1987.
- [17] Stuart Russell, John Binder, Daphne Koller, and Keiji Kanazawa. Local learning in probabilistic networks with hidden variables. In *Proceedings of the 14th international joint conference on Artificial intelligence (IJCAI-95)*, pages 1146–1152. Morgan Kaufmann Publishers Inc. San Francisco, CA, USA, 1995.
- [18] Ilya Shpitser and Judea Pearl. Identification of joint interventional distributions in recursive semi-Markovian causal models. In *National Conference on Artificial Intelligence*, volume 21. AUAI Press, 2006.
- [19] Ilya Shpitser and Judea Pearl. Complete identification methods for the causal hierarchy. *Journal of Machine Learning Research*, 9(Sep):1941–1979, 2008.
- [20] Ilya Shpitser, Thomas S. Richardson, and James M. Robins. An efficient algorithm for computing interventional distributions in latent variable causal models. In *Uncertainty in Artificial Intelligence*, volume 27. AUAI Press, 2011.
- [21] P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. Springer Verlag, New York, 1993.
- [22] Jin Tian and Judea Pearl. On the testable implications of causal models with hidden variables. In *Uncertainty in Artificial Intelligence*, volume 18, pages 519–527. AUAI Press, 2002.
- [23] T. S. Verma and Judea Pearl. Equivalence and synthesis of causal models. Technical Report R-150, Department of Computer Science, University of California, Los Angeles, 1990.

---

# Improved Asymmetric Locality Sensitive Hashing (ALSH) for Maximum Inner Product Search (MIPS)

---

**Anshumali Shrivastava**

Department of Computer Science  
Computing and Information Science  
Cornell University  
Ithaca, NY 14853, USA  
anshu@cs.cornell.edu

**Ping Li**

Department of Statistics and Biostatistics  
Department of Computer Science  
Rutgers University  
Piscataway, NJ 08854, USA  
pingli@stat.rutgers.edu

## Abstract

Recently we showed that the problem of Maximum Inner Product Search (MIPS) is efficient and it admits provably sub-linear hashing algorithms. In [23], we used asymmetric transformations to convert the problem of approximate MIPS into the problem of approximate near neighbor search which can be efficiently solved using L2-LSH. In this paper, we revisit the problem of MIPS and argue that the quantizations used in L2-LSH is suboptimal for MIPS compared to signed random projections (SRP) which is another popular hashing scheme for cosine similarity (or correlations). Based on this observation, we provide different asymmetric transformations which convert the problem of approximate MIPS into the problem amenable to SRP instead of L2-LSH. An additional advantage of our scheme is that we also obtain LSH type space partitioning which is not possible with the existing scheme. Our theoretical analysis shows that the new scheme is significantly better than the original scheme for MIPS. Experimental evaluations strongly support the theoretical findings. In addition, we also provide the first empirical comparison that shows the superiority of hashing over tree based methods [21] for MIPS.

## 1 Introduction

In this paper, we revisit the problem of *Maximum Inner Product Search (MIPS)*, which was studied in our recent work [23]. In this work we present the first provably fast algorithm for MIPS, which was considered hard [21, 15]. Given an input query point  $q \in \mathbb{R}^D$ , the task of MIPS is to find  $p \in \mathcal{S}$ , where  $\mathcal{S}$  is a giant collection of size  $N$ , which maximizes (approximately) the **inner product**  $q^T p$ :

$$p = \arg \max_{x \in \mathcal{S}} q^T x \quad (1)$$

The MIPS problem is related to the problem of *near neighbor search (NNS)*. For example, L2-NNS

$$p = \arg \min_{x \in \mathcal{S}} \|q - x\|_2^2 = \arg \min_{x \in \mathcal{S}} (\|x\|_2^2 - 2q^T x) \quad (2)$$

or, correlation-NNS

$$p = \arg \max_{x \in \mathcal{S}} \frac{q^T x}{\|q\| \|x\|} = \arg \max_{x \in \mathcal{S}} \frac{q^T x}{\|x\|} \quad (3)$$

These three problems are equivalent if the norm of every element  $x \in \mathcal{S}$  is constant. Clearly, the value of the norm  $\|q\|_2$  has no effect for the argmax. In many scenarios, MIPS arises naturally at places where the norms of the elements in  $\mathcal{S}$  have significant variations [15]. As reviewed in our prior work [23], examples of applications of MIPS include recommender system [16, 5, 15], large-scale object detection with DPM [9, 7, 14, 14], structural SVM [7], and multi-class label prediction [21, 15, 25].

**Asymmetric LSH (ALSH):** Locality Sensitive Hashing (LSH) [13] is popular in practice for efficiently solving NNS. In our prior work [23], the concept of “asymmetric LSH” (ALSH) was formalized and one can transform the input query  $Q(p)$  and data in the collection  $P(x)$  independently, where the transformations  $Q$  and  $P$  are different. In [23] we developed a particular set of transformations to convert MIPS into L2-NNS and then solved the problem by standard hashing i.e. L2-LSH [6]. In this paper, we name the scheme in [23] as **L2-ALSH**. Later we showed in [24] the flexibility and the power of the asymmetric framework developed in [23] by constructing a provably superior scheme for binary data. Prior to our work, asymmetry was applied for hashing higher order similarity [22], sketching [8], hashing different subspaces [3], and data dependent hashing [20] which unlike locality sensitive hashing do not come with provable runtime guarantees. Explicitly constructing asymmetric transformation tailored for a particular similarity, given an existing LSH, was the first observation made in [23] due to which MIPS, a sought after problem, became provably fast and practical.

It was argued in [17] that the quantizations used in traditional L2-LSH is suboptimal and it hurts the variance of the hashes. This raises a natural question that L2-ALSH which uses L2-LSH as a subroutine for solving MIPS could be suboptimal and there may be a better hashing scheme. We provide such a scheme in this work.

**Our contribution:** Based on the observation that the quantizations used in traditional L2-LSH is suboptimal, in this study, we propose another scheme for ALSH, by developing a new set of asymmetric transformations to convert MIPS into a problem of correlation-NNS, which is solved by “signed random projections” (SRP) [11, 4]. The new scheme thus avoids the use of L2-LSH. We name this new scheme as **Sign-ALSH**. Our theoretical analysis and experimental study show that Sign-ALSH is more advantageous than L2-ALSH for MIPS.

For inner products asymmetry is unavoidable. In case of L2-ALSH, due to asymmetry, we loose the capability to generate LSH like random data partitions for efficient clustering [12]. We show that for inner products with Sign-ALSH there is a novel formulation that allows us to generate such partitions for inner products. With existing L2-ALSH such formulation does not work.

Apart from providing a better hashing scheme, we also provide comparisons of the Sign-ALSH with cone trees [21]. Our empirical evaluations on three real datasets show that hashing based methods are superior over the tree based space partitioning methods. Since there is no existing comparison of hashing based methods with tree based methods for the problem of MIPS, we believe that the results shown in this work will be very valuable for practitioners.

## 2 Review: Locality Sensitive Hashing (LSH)

The problem of efficiently finding nearest neighbors has been an active research since the very early days of computer science [10]. Approximate versions of the near neighbor search problem [13] were proposed to break the linear query time bottleneck. The following formulation for approximate near neighbor search is often adopted.

**Definition:** (*c*-Approximate Near Neighbor or *c*-NN) Given a set of points in a *D*-dimensional space  $\mathbb{R}^D$ , and parameters  $S_0 > 0$ ,  $\delta > 0$ , construct a data structure which, given any query point *q*, does the following with probability  $1 - \delta$ : if there exists an  $S_0$ -near neighbor of *q* in  $\mathcal{S}$ , it reports some  $cS_0$ -near neighbor of *q* in  $\mathcal{S}$ .

*Locality Sensitive Hashing* (LSH) [13] is a family of functions, with the property that more similar items have a higher collision probability. LSH trades off query time with extra (one time) preprocessing cost and space. Existence of an LSH family translates into provably sublinear query time algorithm for *c*-NN problems.

**Definition:** (Locality Sensitive Hashing (LSH)) A family  $\mathcal{H}$  is called  $(S_0, cS_0, p_1, p_2)$ -sensitive if, for any two points  $x, y \in \mathbb{R}^D$ , *h* chosen uniformly from  $\mathcal{H}$  satisfies:

- if  $Sim(x, y) \geq S_0$  then  $Pr_{\mathcal{H}}(h(x) = h(y)) \geq p_1$
- if  $Sim(x, y) \leq cS_0$  then  $Pr_{\mathcal{H}}(h(x) = h(y)) \leq p_2$

For efficient approximate nearest neighbor search,  $p_1 > p_2$  and  $c < 1$  is needed.

**Fact 1:** Given a family of  $(S_0, cS_0, p_1, p_2)$  -sensitive hash functions, one can construct a data structure for *c*-NN with  $O(n^\rho \log n)$  query time and space  $O(n^{1+\rho})$ , where  $\rho = \frac{\log p_1}{\log p_2} < 1$ .

LSH is a generic framework and an implementation of LSH requires a concrete hash function.

### 2.1 LSH for L2 distance

[6] presented an LSH family for  $L_2$  distances. Formally, given a fixed window size *r*, we sample a random vector *a* with each component from i.i.d. normal, i.e.,  $a_i \sim N(0, 1)$ , and a scalar *b* generated uniformly at random from  $[0, r]$ . The hash function is defined as:

$$h_{a,b}^{L2}(x) = \left\lfloor \frac{a^T x + b}{r} \right\rfloor \quad (4)$$

where  $\lfloor \cdot \rfloor$  is the floor operation. The collision probability under this scheme can be shown to be

$$\begin{aligned} Pr(h_{a,b}^{L2}(x) = h_{a,b}^{L2}(y)) &= 1 - 2\Phi(-r/d) - \frac{2}{\sqrt{2\pi}(r/d)} \left(1 - e^{-(r/d)^2/2}\right) = F_r(d) \end{aligned} \quad (5)$$

where  $\Phi(x) = \int_{-\infty}^x \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx$  and  $d = \|x - y\|_2$  is the Euclidean distance between the vectors *x* and *y*.

### 2.2 LSH for correlation

Another popular LSH family is the so-called “sign random projections” [11, 4]. Again, we choose a random vector *a* with  $a_i \sim N(0, 1)$ . The hash function is defined as:

$$h^{Sign}(x) = sign(a^T x) \quad (6)$$

And collision probability is

$$Pr(h^{Sign}(x) = h^{Sign}(y)) = 1 - \frac{1}{\pi} \cos^{-1} \left( \frac{x^T y}{\|x\| \|y\|} \right) \quad (7)$$

This scheme is known as *signed random projections* (SRP).

## 3 Review of ALSH for MIPS and L2-ALSH

In [23], it was shown that the framework of locality sensitive hashing is restrictive for solving MIPS. The inherent assumption of the same hash function for both the transformation as well as the query was unnecessary in the classical LSH framework and it was the main hurdle in finding provable sub-linear algorithms for MIPS with LSH. For the theoretical guarantees of LSH to work there was no requirement of symmetry. Incorporating asymmetry in the hashing schemes was the key in solving MIPS efficiently.

**Definition [23]:** (*Asymmetric* Locality Sensitive Hashing (ALSH)) A family  $\mathcal{H}$ , along with the two vector functions  $Q : \mathbb{R}^D \mapsto \mathbb{R}^{D'}$  (**Query Transformation**) and  $P :$

$\mathbb{R}^D \mapsto \mathbb{R}^{D'}$  (**Preprocessing Transformation**), is called  $(S_0, cS_0, p_1, p_2)$ -sensitive if for a given  $c$ -NN instance with query  $q$ , and the hash function  $h$  chosen uniformly from  $\mathcal{H}$  satisfies the following:

- if  $Sim(q, x) \geq S_0$  then  $Pr_{\mathcal{H}}(h(Q(q)) = h(P(x))) \geq p_1$
- if  $Sim(q, x) \leq cS_0$  then  $Pr_{\mathcal{H}}(h(Q(q)) = h(P(x))) \leq p_2$

Here  $x$  is any point in the collection  $\mathcal{S}$ .

Note that the query transformation  $Q$  is only applied on the query and the pre-processing transformation  $P$  is applied to  $x \in \mathcal{S}$  while creating hash tables. By letting  $Q(x) = P(x) = x$ , we can recover the vanilla LSH. Using different transformations (i.e.,  $Q \neq P$ ), it is possible to counter the fact that self similarity is not highest with inner products which is the main argument of failure of LSH. We just need the probability of the new collision event  $\{h(Q(q)) = h(P(y))\}$  to satisfy the conditions of definition of ALSH for  $Sim(q, y) = q^T y$ .

**Theorem 1** [23] *Given a family of hash function  $\mathcal{H}$  and the associated query and preprocessing transformations  $P$  and  $Q$ , which is  $(S_0, cS_0, p_1, p_2)$ -sensitive, one can construct a data structure for  $c$ -NN with  $O(n^\rho \log n)$  query time and space  $O(n^{1+\rho})$ , where  $\rho = \frac{\log p_1}{\log p_2}$ .*

[23] provided an explicit construction of ALSH, which we call **L2-ALSH**. Without loss of generality, one can assume

$$\|x_i\|_2 \leq U < 1, \quad \forall x_i \in \mathcal{S} \quad (8)$$

for some  $U < 1$ . If this is not the case, then we can always scale down the norms without altering the  $\arg \max$ . Since the norm of the query does not affect the  $\arg \max$  in MIPS, for simplicity it was assumed  $\|q\|_2 = 1$ . This condition can be removed easily (see Section 5 for details). In L2-ALSH, two vector transformations  $P : \mathbb{R}^D \mapsto \mathbb{R}^{D+m}$  and  $Q : \mathbb{R}^D \mapsto \mathbb{R}^{D+m}$  are defined as follows:

$$P(x) = [x; \|x\|_2^2; \|x\|_2^4; \dots; \|x\|_2^{2^m}] \quad (9)$$

$$Q(x) = [x; 1/2; 1/2; \dots; 1/2], \quad (10)$$

where  $[\cdot]$  is the concatenation.  $P(x)$  appends  $m$  scalars of the form  $\|x\|_2^{2^i}$  at the end of the vector  $x$ , while  $Q(x)$  simply appends  $m$  “1/2” to the end of the vector  $x$ . By observing

$$\begin{aligned} \|P(x_i)\|_2^2 &= \|x_i\|_2^2 + \|x_i\|_2^4 + \dots + \|x_i\|_2^{2^m} + \|x_i\|_2^{2^{m+1}} \\ \|Q(q)\|_2^2 &= \|q\|_2^2 + m/4 = 1 + m/4 \\ Q(q)^T P(x_i) &= q^T x_i + \frac{1}{2}(\|x_i\|_2^2 + \|x_i\|_2^4 + \dots + \|x_i\|_2^{2^m}) \end{aligned}$$

one can obtain the following key equality:

$$\|Q(q) - P(x_i)\|_2^2 = (1 + m/4) - 2q^T x_i + \|x_i\|_2^{2^{m+1}} \quad (11)$$

Since  $\|x_i\|_2 \leq U < 1$ , we have  $\|x_i\|_2^{2^{m+1}} \rightarrow 0$  at the tower rate (exponential to exponential). Thus, as long as  $m$  is not

too small (e.g.,  $m \geq 3$  would suffice), we have

$$\arg \max_{x \in \mathcal{S}} q^T x \simeq \arg \min_{x \in \mathcal{S}} \|Q(q) - P(x)\|_2 \quad (12)$$

This scheme is the first connection between solving unnormalized MIPS and approximate near neighbor search. Transformations  $P$  and  $Q$ , when norms are less than 1, provide correction to the L2 distance  $\|Q(q) - P(x_i)\|_2$  making it rank correlate with the (un-normalized) inner product.

### 3.1 Intuition for the Better Scheme : Why Signed Random Projections (SRP)?

Recently in [17, 18], it was observed that the quantization of random projections used by traditional L2-LSH scheme is not desirable when the data is normalized and in fact the shift  $b$  in Eq. (4) hurts the variance leading to less informative hashes. The sub-optimality of L2-LSH hints towards existence of better hashing functions for MIPS.

As previously argued, when the data are normalized then both L2-NNS and correlation-NNS are equivalent to MIPS. Therefore, for normalized data we can use either L2-LSH which is popular LSH for L2 distance or SRP which is popular LSH for correlation to solve MIPS directly. This raises a natural question “Which will perform better?”

If we assume that the data are normalized, i.e., all the norms are equal to 1, then both SRP and L2-LSH are monotonic in the inner product and their corresponding  $\rho$  values for retrieving max inner product can be computed as

$$\rho_{SRP} = \frac{\log\left(1 - \frac{1}{\pi} \cos^{-1}(S_0)\right)}{\log\left(1 - \frac{1}{\pi} \cos^{-1}(cS_0)\right)} \quad (13)$$

$$\rho_{L2-LSH} = \frac{\log\left(F_r(\sqrt{2-2S_0})\right)}{\log\left(F_r(\sqrt{2-2cS_0})\right)} \quad (14)$$

where the function  $F_r(\cdot)$  is given by Eq. (5). The values of  $\rho_{SRP}$  and  $\rho_{L2-LSH}$  for different  $S_0 = \{0.1, 0.2, \dots, 0.9, 0.95\}$  with respect to approximation ratio  $c$  is shown in Figure 1. We use standard recommendation of  $r = 2.5$  for L2-LSH. We can clearly see that  $\rho_{SRP}$  is consistently better than  $\rho_{L2-LSH}$  given any  $S_0$  and  $c$ . Thus, for MIPS with normalized data L2-LSH type of quantization given by equation 5 seems suboptimal. It is clear that when the data is normalized then SRP is always a better choice for MIPS as compared to L2-LSH. This motivates us to explore the possibility of better hashing algorithm for general (unnormalized) instance of MIPS using SRP, which will have impact in many applications as pointed out in [23].

Asymmetric transformations give us enough flexibility to modify norms without changing inner products. The transformations provided in [23] used this flexibility to convert MIPS to standard near neighbor search in  $L_2$  space for which we have standard hash functions. For binary data, [24] showed a strictly superior construction, the asymmetric minwise hashing, which outperforms all ALSHs made for general MIPS.



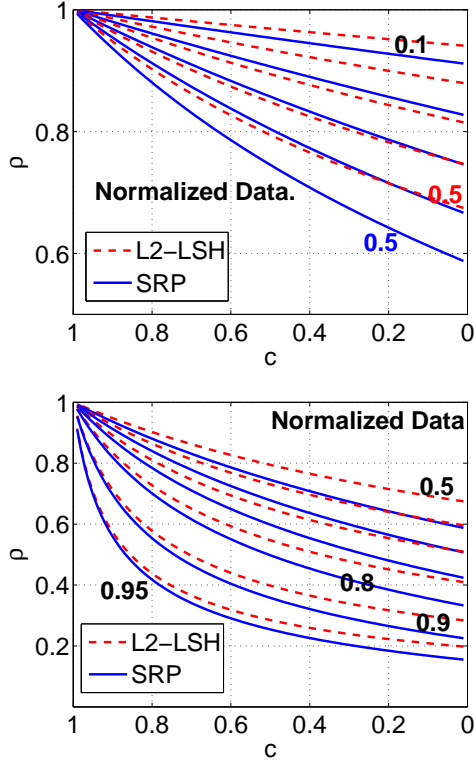


Figure 1: Values of  $\rho_{SRP}$  and  $\rho_{L2-LSH}$  (Lower is better) for normalized data. It is clear that SRP is more suited for retrieving inner products when the data is normalized

Signed random projections are popular hash functions widely adopted for correlation or cosine similarity. We use asymmetric transformations to convert approximate MIPS into approximate maximum correlation search and thus we avoid the use of sub-optimal L2-LSH. The collision probability of the hash functions is one of the key constituents which determine the efficiency of the obtained ALSH algorithm. We show that our proposed transformation with SRP is better suited for ALSH compared to the existing L2-ALSH for solving general MIPS instance.

## 4 The New Proposal: Sign-ALSH

### 4.1 From MIPS to Correlation-NNS

We assume for simplicity that  $\|q\|_2 = 1$  as the norm of the query does not change the ordering, we show in the next section how to get rid of this assumption. Without loss of generality let  $\|x_i\|_2 \leq U < 1, \forall x_i \in \mathcal{S}$  as it can always be achieved by scaling the data by large enough number. We define two vector transformations  $P : \mathbb{R}^D \mapsto \mathbb{R}^{D+m}$  and  $Q : \mathbb{R}^D \mapsto \mathbb{R}^{D+m}$  as follows:

$$P(x) = [x; 1/2 - \|x\|_2^2; 1/2 - \|x\|_2^4; \dots; 1/2 - \|x\|_2^{2^m}] \quad (15)$$

$$Q(x) = [x; 0; 0; \dots; 0], \quad (16)$$

Using  $\|Q(q)\|_2^2 = \|q\|_2^2 = 1$ ,  $Q(q)^T P(x_i) = q^T x_i$ , and

$$\begin{aligned} & \|P(x_i)\|_2^2 \\ &= \|x_i\|_2^2 + 1/4 + \|x_i\|_2^4 - \|x_i\|_2^2 + 1/4 + \|x_i\|_2^8 - \|x_i\|_2^4 + \dots \\ &+ 1/4 + \|x_i\|_2^{2^{m+1}} - \|x_i\|_2^{2^m} \\ &= m/4 + \|x_i\|_2^{2^{m+1}} \end{aligned}$$

we obtain the following key equality:

$$\frac{Q(q)^T P(x_i)}{\|Q(q)\|_2 \|P(x_i)\|_2} = \frac{q^T x_i}{\sqrt{m/4 + \|x_i\|_2^{2^{m+1}}}} \quad (17)$$

The term  $\|x_i\|_2^{2^{m+1}} \rightarrow 0$ , again vanishes at the tower rate. This means we have approximately

$$\arg \max_{x \in \mathcal{S}} q^T x \approx \arg \max_{x \in \mathcal{S}} \frac{Q(q)^T P(x_i)}{\|Q(q)\|_2 \|P(x_i)\|_2} \quad (18)$$

This provides another solution for solving MIPS using known methods for approximate correlation-NNS. Asymmetric transformations  $P$  and  $Q$  provide a lot of flexibility. Note that transformations  $P$  and  $Q$  are not unique for this task and there are other possibilities [2, 19]. It should be further noted that even scaling data and query differently is asymmetry in a strict sense because it changes the distribution of the hashes. Flexibility in choosing the transformations  $P$  and  $Q$  allow us to use signed random projections and thereby making possible to avoid suboptimal L2-LSH.

### 4.2 Fast MIPS Using Sign Random Projections

Eq. (18) shows that MIPS reduces to the standard approximate near neighbor search problem which can be efficiently solved by sign random projections, i.e.,  $h^{Sign}$  (defined by Eq. (6)). Formally, we can state the following theorem.

**Theorem 2** Given a  $c$ -approximate instance of MIPS, i.e.,  $Sim(q, x) = q^T x$ , and a query  $q$  such that  $\|q\|_2 = 1$  along with a collection  $\mathcal{S}$  having  $\|x\|_2 \leq U < 1 \forall x \in \mathcal{S}$ . Let  $P$  and  $Q$  be the vector transformations defined in Eq. (15) and Eq. (16), respectively. We have the following two conditions for hash function  $h^{Sign}$  (defined by Eq. (6))

- if  $q^T x \geq S_0$  then

$$\begin{aligned} & Pr[h^{Sign}(Q(q)) = h^{Sign}(P(x))] \\ & \geq 1 - \frac{1}{\pi} \cos^{-1} \left( \frac{S_0}{\sqrt{m/4 + U^{2^{m+1}}}} \right) \end{aligned}$$

- if  $q^T x \leq cS_0$  then

$$\begin{aligned} & Pr[h^{Sign}(Q(q)) = h^{Sign}(P(x))] \\ & \leq 1 - \frac{1}{\pi} \cos^{-1} \left( \frac{\min\{cS_0, z^*\}}{\sqrt{m/4 + (\min\{cS_0, z^*\})^{2^{m+1}}}} \right) \end{aligned}$$

$$\text{where } z^* = \left( \frac{m/2}{2^{m+1}-2} \right)^{2^{-m-1}}.$$

**Proof:** When  $q^T x \geq S_0$ , we have, according to Eq. (7)

$$\begin{aligned} & Pr[h^{\text{Sign}}(Q(q)) = h^{\text{Sign}}(P(x))] \\ &= 1 - \frac{1}{\pi} \cos^{-1} \left( \frac{q^T x}{\sqrt{m/4 + \|x\|_2^{2^{m+1}}}} \right) \\ &\geq 1 - \frac{1}{\pi} \cos^{-1} \left( \frac{q^T x}{\sqrt{m/4 + U^{2^{m+1}}}} \right) \end{aligned}$$

When  $q^T x \leq cS_0$ , by noting that  $q^T x \leq \|x\|_2$ , we have

$$\begin{aligned} & Pr[h^{\text{Sign}}(Q(q)) = h^{\text{Sign}}(P(x))] \\ &= 1 - \frac{1}{\pi} \cos^{-1} \left( \frac{q^T x}{\sqrt{m/4 + \|x\|_2^{2^{m+1}}}} \right) \\ &\leq 1 - \frac{1}{\pi} \cos^{-1} \left( \frac{q^T x}{\sqrt{m/4 + (q^T x)^{2^{m+1}}}} \right) \end{aligned}$$

For this one-dimensional function  $f(z) = \frac{z}{\sqrt{a+z^b}}$ , where  $z = q^T x$ ,  $a = m/4$  and  $b = 2^{m+1} \geq 2$ , we know

$$f'(z) = \frac{a - z^b (b/2 - 1)}{(a + z^b)^{3/2}}$$

One can also check that  $f''(z) \leq 0$  for  $0 < z < 1$ , i.e.,  $f(z)$  is a concave function. The maximum of  $f(z)$  is attained at  $z^* = \left(\frac{2a}{b-2}\right)^{1/b} = \left(\frac{m/2}{2^{m+1}-2}\right)^{2^{-m-1}}$ . If  $z^* \geq cS_0$ , then we need to use  $f(cS_0)$  as the bound.  $\square$

Therefore, we have obtained, in LSH terminology,

$$p_1 = 1 - \frac{1}{\pi} \cos^{-1} \left( \frac{S_0}{\sqrt{m/4 + U^{2^{m+1}}}} \right) \quad (19)$$

$$p_2 = 1 - \frac{1}{\pi} \cos^{-1} \left( \frac{\min\{cS_0, z^*\}}{\sqrt{m/4 + (\min\{cS_0, z^*\})^{2^{m+1}}}} \right), \quad (20)$$

$$z^* = \left(\frac{m/2}{2^{m+1}-2}\right)^{2^{-m-1}} \quad (21)$$

Theorem 1 allows us to construct data structures with worst case  $O(n^\rho \log n)$  query time guarantees for  $c$ -approximate MIPS, where  $\rho = \frac{\log p_1}{\log p_2}$ . For any given  $c < 1$ , there always exist  $U < 1$  and  $m$  such that  $\rho < 1$ . This way, we obtain a sublinear query time algorithm for MIPS. Because  $\rho$  is a function of 2 parameters, the best query time chooses  $U$  and  $m$ , which minimizes the value of  $\rho$ . For convenience, we define

$$\rho^* = \min_{U, m} \frac{\log \left( 1 - \frac{1}{\pi} \cos^{-1} \left( \frac{S_0}{\sqrt{m/4 + U^{2^{m+1}}}} \right) \right)}{\log \left( 1 - \frac{1}{\pi} \cos^{-1} \left( \frac{\min\{cS_0, z^*\}}{\sqrt{m/4 + (\min\{cS_0, z^*\})^{2^{m+1}}}} \right) \right)} \quad (22)$$

See Figure 2 for the plots of  $\rho^*$ , which also compares the optimal  $\rho$  values for L2-ALSH in the prior work [23]. The results show that Sign-ALSH is noticeably better.

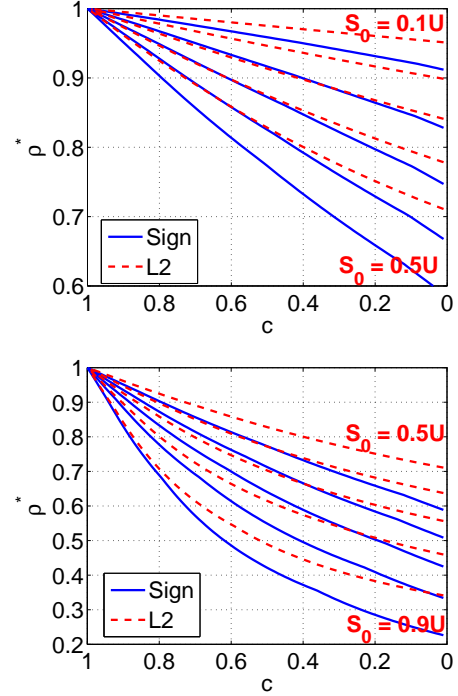


Figure 2: Optimal values of  $\rho^*$  (lower is better) with respect to approximation ratio  $c$  for different  $S_0$ , obtained by a grid search over parameters  $U$  and  $m$ , given  $S_0$  and  $c$ . The curves show that Sign-ALSH (solid curves) is noticeably better than L2-ALSH (dashed curves) in terms of their optimal  $\rho^*$  values. The results for L2-ALSH were from the prior work [23]. For clarity, the results are in two figures.

### 4.3 Parameter Selection

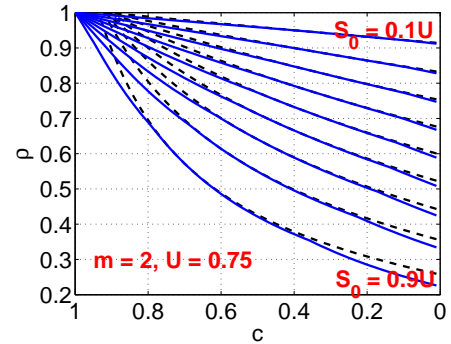


Figure 3: The solid curves are the optimal  $\rho$  values of Sign-ALSH from Figure 2. The dashed curves represent the  $\rho$  values for fixed parameters:  $m = 2$  and  $U = 0.75$  (left panel). Even with fixed parameters, the  $\rho$  does not degrade.

Figure 3 presents the  $\rho$  values for  $(m, U) = (2, 0.75)$ . We can see that even if we use fixed parameters, the per-

formance would only degrade little. This essentially frees practitioners from the burden of choosing parameters.

## 5 Removing Dependency on Norm of Query

Changing norms of the query does not affect the  $\arg \max_{x \in \mathcal{C}} q^T x$ , and hence, in practice for retrieving top- $k$ , normalizing the query should not affect the performance. But for theoretical purposes, we want the runtime guarantee to be independent of  $\|q\|_2$ . Note, both LSH and ALSH schemes solve the  $c$ -approximate instance of the problem, which requires a threshold  $S_0 = q^T x$  and an approximation ratio  $c$ . These quantities change if we change the norms. We can use the same idea used in [23] to get rid of the norm of  $q$ . Transformations  $P$  and  $Q$  were precisely meant to remove the dependency of correlation on the norms of  $x$  but at the same time keeping the inner products same. Let  $M$  be the upper bound on all the norms i.e.  $M = \max_{x \in \mathcal{C}} \|x\|_2$ . In other words  $M$  is the radius of the space.

Let  $U < 1$ , define the transformations,  $T : \mathbb{R}^D \rightarrow \mathbb{R}^D$  as

$$T(x) = \frac{Ux}{M} \quad (23)$$

and transformations  $P, Q : \mathbb{R}^D \rightarrow \mathbb{R}^{D+m}$  are the same for the Sign-ALSH scheme as defined in Eq (15) and (16).

Given the query  $q$  and any data point  $x$ , observe that the inner products between  $P(Q(T(q)))$  and  $Q(P(T(x)))$  is

$$P(Q(T(q)))^T Q(P(T(x))) = q^T x \times \left( \frac{U^2}{M^2} \right) \quad (24)$$

$P(Q(T(q)))$  appends first  $m$  zeros components to  $T(q)$  and then  $m$  components of the form  $1/2 - \|q\|_2^2$ .  $Q(P(T(x)))$  does the same thing but in a different order. Now we are working in  $D + 2m$  dimensions. It is not difficult to see that the norms of  $P(Q(T(q)))$  and  $Q(P(T(x)))$  is given by

$$\|P(Q(T(q)))\|_2 = \sqrt{\frac{m}{4} + \|T(q)\|_2^{2m+1}} \quad (25)$$

$$\|Q(P(T(x)))\|_2 = \sqrt{\frac{m}{4} + \|T(x)\|_2^{2m+1}} \quad (26)$$

The transformations are very asymmetric but we know that it is necessary.

Therefore the correlation or the cosine similarity between  $P(Q(T(q)))$  and  $Q(P(T(x)))$  is

$$Corr = \frac{q^T x \times \left( \frac{U^2}{M^2} \right)}{\sqrt{\frac{m}{4} + \|T(q)\|_2^{2m+1}} \sqrt{\frac{m}{4} + \|T(x)\|_2^{2m+1}}} \quad (27)$$

Note  $\|T(q)\|_2^{2m+1}, \|T(x)\|_2^{2m+1} \leq U < 1$ , therefore both  $\|T(q)\|_2^{2m+1}$  and  $\|T(x)\|_2^{2m+1}$  converge to zero at a tower rate and we get approximate monotonicity of correlation

with the inner products. We can apply sign random projections to hash  $P(Q(T(q)))$  and  $Q(P(T(x)))$ .

As  $0 \leq \|T(q)\|_2^{2m+1} \leq U$  and  $0 \leq \|T(x)\|_2^{2m+1} \leq U$ , it is not difficult to get  $p_1$  and  $p_2$  for Sign-ALSH, without conditions on any norms. Simplifying the expression, we get the following value of optimal  $\rho_u$  ( $u$  for unrestricted).

$$\rho_u^* = \min_{U, m} \frac{\log \left( 1 - \frac{1}{\pi} \cos^{-1} \left( \frac{S_0 \times \left( \frac{U^2}{M^2} \right)}{\frac{m}{4} + U^{2m+1}} \right) \right)}{\log \left( 1 - \frac{1}{\pi} \cos^{-1} \left( \frac{c S_0 \times \left( \frac{4U^2}{M^2} \right)}{m} \right) \right)} \quad (28)$$

$$s.t. \quad U^{2m+1} < \frac{m(1-c)}{4c}, \quad m \in \mathbb{N}^+, \quad \text{and } 0 < U < 1.$$

With this value of  $\rho_u^*$ , we can state our main theorem.

**Theorem 3** *For the problem of  $c$ -approximate MIPS in a bounded space, one can construct a data structure having  $O(n^{\rho_u^*} \log n)$  query time and space  $O(n^{1+\rho_u^*})$ , where  $\rho_u^* < 1$  is the solution to constraint optimization (28).*

Note, for all  $c < 1$ , we always have  $\rho_u^* < 1$  because the constraint  $U^{2m+1} < \frac{m(1-c)}{4c}$  is always true for big enough  $m$ . The only assumption for efficiently solving MIPS that we need is that the space is bounded, which is always satisfied for any finite dataset.  $\rho_u^*$  depends on  $M$ , the radius of the space, which is expected.

## 6 Random Space Partitioning for Inner Product

In this section, we show that due to the nature of the new transformations  $P$  and  $Q$  there is one subtle but surprising advantage of Sign-ALSH over L2-ALSH.

One popular application of LSH (Locality Sensitive Hashing) is random partitioning of the data for large scale clustering, where similar points map to the same partition (or bucket). Such partitions are very useful in many applications [12]. With classical LSH, we simply use  $h(x)$  to generate partition for  $x$ . Since  $Pr_{\mathcal{H}}(h(x) = h(y))$  is high if  $sim(x, y)$  is high, similar points are likely to go into the same partition under the usual LSH mapping. For general ALSH, this property is lost because of asymmetry.

In case of ALSH, we only know that  $Pr(h(P(x)) = h(Q(y)))$  is high if  $sim(x, y)$  is high. Therefore, given  $x$  we cannot determine whether to assign partition using  $h(P(\cdot))$  or  $h(Q(\cdot))$ . Neither  $Pr(h(P(x)) = h(P(y)))$  nor  $Pr_{\mathcal{H}}(h(Q(x)) = h(Q(y)))$  strictly indicates high value of  $sim(x, y)$  in general. Therefore, partitioning property of classical LSH does not hold anymore with general ALSHs. However for the case of inner products using Sign-ALSH, there is a subtle observation which allows us to construct the required assignment function, where pairs of points with high inner products are more likely to get mapped in

the same partition while pairs with low inner products are more likely to map into different partitions.

In case of Sign-ALSH for MIPS, we have the transformations  $P(Q(T(x)))$  and  $Q(P(T(x)))$  given by

$$P(Q(T(x))) = [x; 1/2 - \|T(x)\|_2^2; \dots; 1/2 - \|T(x)\|_2^{2m}, 0, \dots, 0]$$

$$Q(P(T(x))) = [x; 0, \dots, 0, 1/2 - \|T(x)\|_2^2; \dots; 1/2 - \|T(x)\|_2^{2m}].$$

After this transformation, we multiply the generated  $D + 2m$  dimensional vector by a random vector  $a \in \mathbb{R}^{D+2m}$  whose entries are i.i.d. Gaussian followed by taking the sign. For illustration let  $a = [w; s_1, \dots, s_m, t_1, \dots, t_m]$  where  $w \in \mathbb{R}^D$ ,  $b_i$  and  $c_i$  are numbers. All components of  $a$  are i.i.d. from  $N(0, 1)$ . With this notation, we can write the final Sign-ALSH as

$$h^{Sign}(P(Q(T(x)))) = Sign(w^T T(x) + \sum_{i=1}^m s_i (1/2 - \|T(x)\|_2^{2i}))$$

$$h^{Sign}(Q(P(T(x)))) = Sign(w^T T(x) + \sum_{i=1}^m t_i (1/2 - \|T(x)\|_2^{2i}))$$

The key observation here is that  $h^{Sign}(P(Q(T(x))))$  does not depend on  $t_i$  and  $h^{Sign}(Q(P(T(x))))$  does not depend on  $s_i$ . If we define

$$h_w(x) = Sign(w^T T(x) + \sum_{i=1}^m \alpha_i (1/2 - \|T(x)\|_2^{2i})) \quad (29)$$

where  $\alpha_i$  are sampled i.i.d. from  $N(0, 1)$  for every  $x$  independently of everything else. Then, **under the randomization of  $w$** , it is not difficult to show that

$$Pr_w(h_w(x) = h_w(y)) = Pr(h^{Sign}(P(x)) = h^{Sign}(Q(y)))$$

for any  $x, y$ . The term  $Pr(h^{Sign}(P(x)) = h^{Sign}(Q(y)))$  satisfies the LSH like property and therefore, in any partitions using  $h_w$ , points with high inner products are more likely to be together. Thus,  $h_w(x)$  is the required assignment. Note,  $h_w$  is not technically an LSH because we are randomly sampling  $\alpha_i$  for all  $x$  independently. The construction of  $h_w$  using independent randomizations could be of separate interest. To the best of our knowledge, this is the first example of LSH like partition using hash function with independent randomization for every data point.

The function  $h_w$  is little subtle here, we sample  $w$  i.i.d from Gaussian and use the same  $w$  for all  $x$ , but while computing  $h_w$  we use  $\alpha_i$  independent of everything for every  $x$ . The probability is under the randomization of  $w$  and independence of all  $\alpha_i$  ensures the asymmetry. We are not sure if such construction is possible with L2-ALSH. For LSH partitions with binary data, the idea used here can be applied on asymmetric minwise hashing [24].

## 7 Ranking Evaluations

In [23], the L2-ALSH scheme was shown to outperform other reasonable heuristics in retrieving maximum inner products. Since our proposal is an improvement over L2-ALSH, in this section we first present comparisons with L2-ALSH, in particular on ranking experiments.

### 7.1 Datasets

We use three publicly available dataset MNIST, WEBSPAM and RCV1 for evaluations. For each of the three dataset we generate two independent partitions, the query set and the train set. Each element in the query set is used for querying, while the training set serves as the collection  $\mathcal{C}$  that will be searched for MIPS. The statistics of the dataset and the partitions are summarized in Table 1

| Dataset | Dimension  | Query size | Train size |
|---------|------------|------------|------------|
| MNIST   | 784        | 10,000     | 60,000     |
| WEBSPAM | 16,609,143 | 5,000      | 100,000    |
| RCV1    | 47,236     | 5,000      | 100,000    |

Table 1: Datasets used for evaluations.

### 7.2 Evaluations

In this section, we show how the ranking of the two ALSH schemes, L2-ALSH and Sign-ALSH, correlates with inner products. Given a query vector  $q$ , we compute the top-10 gold standard elements based on the actual inner products  $q^T x$ ,  $\forall x \in \mathcal{C}$ , here our collection is the train set. We then generate  $K$  different hash codes of the query  $q$  and all the elements  $x \in \mathcal{C}$  and then compute

$$Matches_x = \sum_{t=1}^K \mathbf{1}(h_t(Q(q)) = h_t(P(x))), \quad (30)$$

where  $\mathbf{1}$  is the indicator function and the subscript  $t$  is used to distinguish independent draws of  $h$ . Based on  $Matches_x$  we rank all the elements  $x$ . Ideally, for a better hashing scheme,  $Matches_x$  should be higher for element  $x$  having higher inner products with the given query  $q$ . This procedure generates a sorted list of all the items for a given query vector  $q$  corresponding to the each of the two asymmetric hash functions under consideration.

For L2-ALSH, we used the same parameters used and recommended in [23]. For Sign-ALSH, we used the recommended choice shown in Section 4.3, which is  $U = 0.75$ ,  $m = 2$ . Note that Sign-ALSH does not have parameter  $r$ .

We compute precision and recall of the top-10 gold standard elements, obtained from the sorted list based on  $Matches_x$ . To compute this precision and recall, we start at the top of the ranked item list and walk down in order. Suppose we are at the  $k^{th}$  ranked item, we check if this element belongs to the gold standard top-10 list. If it is one of the top-10 gold standard elements, then we increment the count of *relevant seen* by 1, else we move to  $k + 1$ . By  $k^{th}$  step, we have already seen  $k$  elements, so the *total items seen* is  $k$ . The precision and recall at that point are

$$Precision = \frac{\text{relevant seen}}{k}, \quad Recall = \frac{\text{relevant seen}}{10}$$

We show performance for  $K \in \{64, 128, 256, 512\}$ . Note that it is important to balance both precision and recall. The

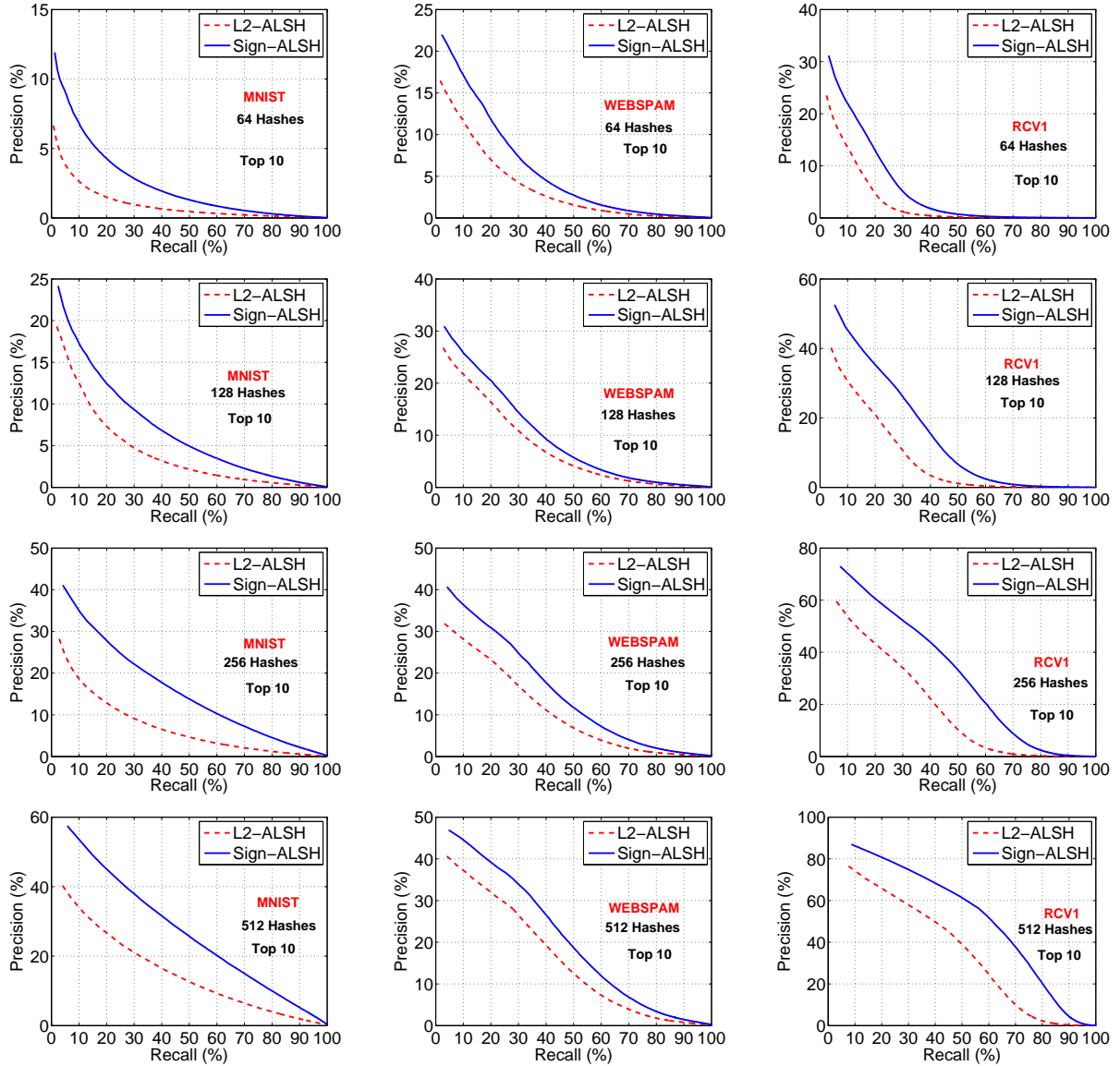


Figure 4: Precision-Recall curves (higher is better). We compare L2-ALSH (using parameters recommended in [23]) with our proposed Sign-ALSH using ( $m = 2, U = 0.75$ ) for retrieving top-10 elements. Sign-ALSH is noticeably better.

method which obtains higher precision at a given recall is superior. Higher precision indicates higher ranking of the top-10 inner products which is desirable. We report averaged precisions and recalls.

The plots for all the three datasets are shown in Figure 4. We can clearly see, that our proposed Sign-ALSH scheme gives significantly higher precision recall curves than the L2-ALSH scheme, indicating better correlation of top inner products with Sign-ALSH compared to L2-ALSH. The results are consistent across datasets.

## 8 Comparisons of Hashing Based and Tree Based Methods for MIPS

We have shown in the previous Section that Sign-ALSH outperforms L2-ALSH in ranking evaluations. In this Section, we consider the actual task of finding the maximum

inner product. Our aim is to estimate the computational saving, in finding the maximum inner product, with Sign-ALSH compared to the existing scheme L2-ALSH. In addition to L2-ALSH which is a hashing scheme, there is another tree based space partitioning method [21] for solving MIPS. Although, in theory, it is known that tree based methods perform poorly [25] due to their exponential dependence on the dimensionality, it is still important to understand the impact of such dependency in practice. Unfortunately no empirical comparison between hashing and tree based methods exists for the problem of MIPS in the literature. To provide such a comparison, we also consider tree based space partitioning method [21] for evaluations. We use the same three datasets as described in Section 7.1.

Tree based and hashing based methodologies are very different in nature. The major difference is in the stopping

criteria. Hashing based methods create buckets and stop the search once they find a good enough point, they may not succeed with some probability. On the other hand, tree based methods use branch and bound criteria to stop exploring further. So it is possible that a tree based algorithm finds the optimal point but continues to explore further requiring more computations. The usual stopping criteria thus makes tree based methods unnecessarily expensive compared to hashing based methods where the criteria is to stop after finding a good point. Therefore, to ensure fair comparisons, we allow the tree based method to stop the evaluations immediately once the algorithm finds the maximum inner product and prevent it from exploring further. Also, in case when hashing based algorithm fails to find the best inner product we resort to the full linear scan and penalize the hashing based algorithm for not succeeding. All this is required to ensure that tree based algorithm is not at any disadvantage compare to hashing methods.

We implemented the bucketing scheme with Sign-ALSH and L2-ALSH. The bucketing scheme requires creating many hash tables during the preprocessing stage. During query phase, given a query, we compute many hashes of the query and probe appropriate buckets in each table. Please refer [1] for more details on the process. We use the same fixed parameters for all the evaluations, i.e., ( $m=2$ ,  $U=0.75$ ) for Sign-ALSH and ( $m=3$ ,  $U=0.83$ ,  $r=2.5$ ) for L2-ALSH as recommended in [23]. The total number of inner products evaluated by a hashing scheme, for a given query, is the total number of hash computation for the query plus the total number of points retrieved from the hash tables. In rare cases, with very small probability, if the hash tables are unable to retrieve the gold standard maximum inner product, we resort to linear scan and also include the total number of inner products computed during the linear scan. We stop as soon as we reach the gold standard point.

We implemented Algorithm 5 from [21], which is the best performing algorithm as shown in the evaluations. For this algorithm, we need to select one parameter which is the minimum number of elements in the node required for splitting. We found that on all the three datasets the value of 100 for this parameter works the best among {500, 200, 100, 50}. Therefore, we use 100 in all our experiments. The total number of inner products evaluated by tree based algorithm is the total number of points reported plus the total number of nodes visited, where we compute the branch and bound constraint. Again we stop the search process as soon as we reach the point with gold standard maximum inner product. As argued, we need this common stopping condition to compare with hashing based methods, where we do not have any other stopping criteria [13].

For every query we compute the number of inner products evaluated by different methods for MIPS. We report the mean of the total number of inner products evaluated per query in Table 2. We can clearly see that hashing based

|          | Sign-ALSH    | L2-ALSH | Cone Trees |
|----------|--------------|---------|------------|
| MNIST    | <b>7,944</b> | 9,971   | 11,202     |
| WEBS-PAM | <b>2,866</b> | 3,813   | 22,467     |
| RCV1     | <b>9,951</b> | 11,883  | 38,162     |

Table 2: Average number of inner products evaluated per query by different MIPS algorithms. Both Sign-ALSH and L2-ALSH [23] outperform cone trees [21]. Sign-ALSH is always superior compared to L2-ALSH for MIPS.

methods are always better than the tree based algorithm. Except on MNIST dataset, hashing based methods are significantly superior, which is also not surprising because MNIST is an image dataset having low intrinsic dimensionality. Among the two hashing schemes Sign-ALSH is always better than L2-ALSH, which verifies our theoretical findings and supports our arguments in favor of Sign-ALSH over L2-ALSH for MIPS.

## 9 Conclusion

The MIPS (maximum inner product search) problem has numerous important applications in machine learning, databases, and information retrieval. [23] developed the framework of Asymmetric LSH and provided an explicit scheme (L2-ALSH) for approximate MIPS in sublinear time. L2-ALSH uses L2-LSH as a subroutine which uses suboptimal quantizations. In this study, we present another asymmetric transformation scheme (Sign-ALSH) which converts the problem of maximum inner products into the problem of maximum correlation search, which is subsequently solved by sign random projections, thereby avoiding the use of L2-LSH.

Theoretical analysis and experimental study demonstrate that **Sign-ALSH** can be noticeably more advantageous than **L2-ALSH**. The new transformations with Sign-ALSH can be adapted to generate LSH like random data partitions which is very useful for large scale clustering. Such an adaptation is not possible with existing L2-ALSH. This was a rather unexpected advantage of the proposed Sign-ALSH over L2-ALSH. We also establish by experiments that hashing based algorithms are superior to tree based space partitioning methods for MIPS.

It should be noted that for MIPS over binary data our recent work asymmetric minwise hashing [24] should be used. We showed that for binary domain asymmetric minwise hashing is both empirically and provably superior, please see [24] for more details.

## 10 Acknowledgement

The work is partially supported by NSF-III-1360971, NSF-Bigdata-1419210, ONR-N00014-13-1-0764, and AFOSR-FA9550-13-1-0137. We would like to thank the reviewers of AISTATS 2015 and UAI 2015. We also thank Sanjiv Kumar and Hadi Daneshmand for pleasant discussions.

## References

- [1] A. Andoni and P. Indyk. E2lsh: Exact euclidean locality sensitive hashing. Technical report, 2004.
- [2] Y. Bachrach, Y. Finkelstein, R. Gilad-Bachrach, L. Katzir, N. Koenigstein, N. Nice, and U. Paquet. Speeding up the xbox recommender system using a euclidean transformation for inner-product spaces. In *Proceedings of the 8th ACM Conference on Recommender Systems*, RecSys '14, 2014.
- [3] R. Basri, T. Hassner, and L. Zelnik-Manor. Approximate nearest subspace search with applications to pattern recognition. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.
- [4] M. S. Charikar. Similarity estimation techniques from rounding algorithms. In *STOC*, pages 380–388, Montreal, Quebec, Canada, 2002.
- [5] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 39–46. ACM, 2010.
- [6] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni. Locality-sensitive hashing scheme based on  $p$ -stable distributions. In *SCG*, pages 253 – 262, Brooklyn, NY, 2004.
- [7] T. Dean, M. A. Ruzon, M. Segal, J. Shlens, S. Vijayanarasimhan, and J. Yagnik. Fast, accurate detection of 100,000 object classes on a single machine. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1814–1821. IEEE, 2013.
- [8] W. Dong, M. Charikar, and K. Li. Asymmetric distance estimation with sketches for similarity search in high-dimensional spaces. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 123–130. ACM, 2008.
- [9] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(9):1627–1645, 2010.
- [10] J. H. Friedman and J. W. Tukey. A projection pursuit algorithm for exploratory data analysis. *IEEE Transactions on Computers*, 23(9):881–890, 1974.
- [11] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of ACM*, 42(6):1115–1145, 1995.
- [12] T. H. Haveliwala, A. Gionis, and P. Indyk. Scalable techniques for clustering the web. In *WebDB*, pages 129–134, 2000.
- [13] P. Indyk and R. Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *STOC*, pages 604–613, Dallas, TX, 1998.
- [14] T. Joachims, T. Finley, and C.-N. J. Yu. Cutting-plane training of structural svms. *Machine Learning*, 77(1):27–59, 2009.
- [15] N. Koenigstein, P. Ram, and Y. Shavitt. Efficient retrieval of recommendations in a matrix factorization framework. In *CIKM*, pages 535–544, 2012.
- [16] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems.
- [17] P. Li, M. Mitzenmacher, and A. Shrivastava. Coding for random projections. In *ICML*, 2014.
- [18] P. Li, M. Mitzenmacher, and A. Shrivastava. Coding for random projections and approximate near neighbor search. Technical report, arXiv:1403.8144, 2014.
- [19] B. Neyshabur and N. Srebro. On symmetric and asymmetric lshs for inner product search. Technical report, arXiv:1410.5518, 2014.
- [20] B. Neyshabur, N. Srebro, R. R. Salakhutdinov, Y. Makarychev, and P. Yadollahpour. The power of asymmetry in binary hashing. In *Advances in Neural Information Processing Systems*, pages 2823–2831, 2013.
- [21] P. Ram and A. G. Gray. Maximum inner-product search using cone trees. In *KDD*, pages 931–939, 2012.
- [22] A. Shrivastava and P. Li. Beyond pairwise: Provably fast algorithms for approximate k-way similarity search. In *NIPS*, Lake Tahoe, NV, 2013.
- [23] A. Shrivastava and P. Li. Asymmetric lsh (alsh) for sublinear time maximum inner product search (mips). In *NIPS*, Montreal, CA, 2014.
- [24] A. Shrivastava and P. Li. Asymmetric minwise hashing for indexing binary inner products and set containment. In *WWW*, 2015.
- [25] R. Weber, H.-J. Schek, and S. Blott. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In *Proceedings of the 24rd International Conference on Very Large Data Bases, VLDB '98*, pages 194–205, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

---

# Learning Optimal Chain Graphs with Answer Set Programming

---

**Dag Sonntag**  
ADIT, IDA,  
Linköping University

**Matti Järvisalo**  
HIIT, Dept. Comp. Sci.,  
University of Helsinki

**Jose M. Peña**  
ADIT, IDA,  
Linköping University

**Antti Hyttinen**  
HIIT, Dept. Comp. Sci.,  
University of Helsinki

## Abstract

Learning an optimal chain graph from data is an important hard computational problem. We present a new approach to solve this problem for various objective functions without making any assumption on the probability distribution at hand. Our approach is based on encoding the learning problem declaratively using the answer set programming (ASP) paradigm. Empirical results show that our approach provides at least as accurate solutions as the best solutions provided by the existing algorithms, and overall provides better accuracy than any single previous algorithm.

## 1 INTRODUCTION

Learning an optimal structure for a graphical model is a well-known and important hard computational problem. Indeed, various learning algorithms have been proposed over the years for different classes of graphical models. These algorithms can be categorized into in-exact (often local search style) approaches and exact approaches. The algorithms in the former category typically scale better, but are not in general guaranteed to produce optimal solutions without restrictive assumptions on the probability distribution at hand. The algorithms in the latter category, due to the NP-hardness of the underlying optimization problem, require more computational resources, but in turn can provide optimal solutions in much more general settings.

Two important, widely studied and applied classes of probabilistic graphical models are Bayesian networks (whose structure is represented by directed acyclic graphs), and Markov networks (represented by undirected graphs). In this paper we focus on *chain graphs* (CGs), a superclass of both Bayesian and Markov networks. CGs are hybrid graphs that can contain both directed and undirected edges, but are not allowed to contain semidirected cycles. This

makes CGs more expressive than Bayesian and Markov networks in the sense that CGs can model both symmetric (like Markov networks) and asymmetric (like Bayesian networks) relations between random variables, and hence allow for a wider range of independence models to be represented. For example, for 20 random variables, CGs can represent approximately 1000 times as many models as Bayesian networks (Sonntag et al., 2015). There exist multiple interpretations of CGs in the literature. Here we focus on the classical LWF interpretation by Frydenberg (1990); Lauritzen and Wermuth (1989).

In this work we take on the challenging task of developing exact structure learning algorithms for CGs. To our best knowledge, to date only three learning algorithms CGs have been proposed: PC (Studený, 1997), LCD (Ma et al., 2008), and CKES (Peña et al., 2014). Each of these algorithms implement forms of local search, and are guaranteed to find (inclusion) optimal solutions only under restrictive assumptions on the input. Specifically, PC and LCD assume that the probability distribution at hand is faithful to a CG. The CKES algorithm, on the other hand, assumes that the probability distribution at hand satisfies the so-called composition property. While this assumption is considerably weaker than the faithfulness assumption, it leaves room for developing CG learning algorithms that can provide optimal solutions in more general settings.

The main contribution of this paper is a versatile approach to learning CGs. Our approach improves on the earlier CG learning algorithms in that it is guaranteed to find (e.g., inclusion) optimal CGs without making assumptions on the probability distribution at hand. Moreover, our approach can easily be adapted to produce optimal CGs wrt other objective functions such as CGs that represent the largest number of independencies, or CGs with the least number of edges. Our approach is based on encoding the CG learning problem in a modular way using the answer set programming (ASP) paradigm. This enables the use of recent advances in state-of-the-art exact ASP optimization solvers that allow for a complete search in the space of CGs. Moreover, due to the expressive constraint modelling



language offered by ASP, our approach allows for integrating user knowledge into the search. This can, for example, be deployed to enforce certain substructures in the produced solutions or to adopt advanced objective functions taking non-binary information about independence constraints into account.

We also present results from an empirical comparison of our approach with the existing CG learning algorithms. As the existing algorithms make assumptions about the probability distribution at hand, the experiments involve only probability distributions that satisfy the assumptions of the existing algorithms (although unnecessary for our approach) in order to avoid biasing the results in our favour. Even in such restricted settings, the results of the evaluation indicate that our approach provides at least as accurate solutions as the best solutions provided by the existing algorithms, and overall provides better accuracy than any single previous algorithm.

This work is motivated by recent work on harnessing declarative programming (including ASP, Boolean satisfiability, maximum satisfiability, and integer programming solvers) to learn optimal Bayesian networks (Cussens and Bartlett, 2013; Berg et al., 2014; Parviainen et al., 2014), Markov networks (Corander et al., 2013) and causal structures (Hyttinen et al., 2013, 2014). However, to our best knowledge, this work is the first investigation into learning optimal CGs with declarative programming.

The rest of this paper is organized as follows. We start by reviewing CGs, their semantics, and the three existing CG learning algorithms (Section 2). We then continue by presenting and extending a set of inference rules for computing separations and non-separations in CGs (Section 3). In Section 4 we present an implementation of the CG learning approach, using the rules from the previous section. This implementation is then evaluated wrt the existing CG learning algorithms in Section 5.

## 2 CHAIN GRAPHS

In this section, we review concepts related to chain graphs as relevant to this work.

Unless otherwise stated, all the graphs in this paper are defined over a finite set of  $N$  nodes. Moreover, the graphs are *simple*, i.e., contain at most one edge between any pair of nodes. The elements of  $N$  are not distinguished from singletons. We consider graphs which may contain both undirected and directed edges. For a given graph  $G$ , we use  $x_1 - x_2$  (resp.,  $x_1 \rightarrow x_2$ ) to denote that  $G$  contains an undirected (resp., directed edge) between two nodes  $x_1$  and  $x_2$ .

The *skeleton* of  $G$  is the undirected graph that has the same adjacencies as  $G$ . A *route* between a node  $x_1$  and a node  $x_n$  in  $G$  is a sequence of (not necessarily distinct) nodes

$x_1, \dots, x_n$  such that  $x_i \rightarrow x_{i+1}$ ,  $x_i \leftarrow x_{i+1}$ , or  $x_i - x_{i+1}$  for all  $1 \leq i < n$ . A route  $x_1, \dots, x_n$  in  $G$  is a *semidirected cycle* if (i)  $x_n = x_1$ , (ii)  $x_1 \rightarrow x_2$  is in  $G$ , and (iii)  $x_i \rightarrow x_{i+1}$  or  $x_i - x_{i+1}$  is in  $G$  for all  $1 < i < n$ . A *chain graph* (CG) is a graph that contains no semidirected cycles.

A *section* of a route  $\rho$  in a CG is a *maximal* (wrt set inclusion) undirected subroute of  $\rho$ . A section  $x_2 - \dots - x_{n-1}$  of  $\rho$  is a *collider section* of  $\rho$  if  $x_1 \rightarrow x_2 - \dots - x_{n-1} \leftarrow x_n$  is a subroute of  $\rho$ . Moreover,  $\rho$  is *Z-open* with  $Z \subseteq N$  if (i) every collider section of  $\rho$  has a node in  $Z$ , and (ii) no non-collider section of  $\rho$  has a node in  $Z$ . Let  $X, Y$  and  $Z$  denote three disjoint subsets of  $N$ . If there is no  $Z$ -open route in a CG  $G$  between nodes in  $X$  and nodes in  $Y$ ,  $X$  is *separated* from  $Y$  given  $Z$  in  $G$ , denoted by  $X \perp_G Y | Z$ , meaning that  $X$  and  $Y$  are represented as conditionally independent given  $Z$  in  $G$ . Otherwise,  $X$  is *non-separated* from  $Y$  given  $Z$  in  $G$ , denoted by  $X \not\perp_G Y | Z$ . An independence model  $M$  is a set of statements of the form  $X \perp_M Y | Z$ , meaning that  $X$  is independent of  $Y$  given  $Z$ . The *independence model* represented by  $G$ , denoted by  $I(G)$ , is the set of represented independencies  $X \perp_G Y | Z$ . We denote by  $X \perp_p Y | Z$  (resp.  $X \not\perp_p Y | Z$ ) that  $X$  is independent (resp. dependent) of  $Y$  given  $Z$  under a probability distribution  $p$ . The independence model induced by  $p$ , denoted by  $I(p)$ , is the set of statements  $X \perp_p Y | Z$ . A distribution  $p$  is *faithful* to a CG  $G$  when  $X \perp_p Y | Z$  iff  $X \perp_G Y | Z$  for all pairwise disjoint subsets  $X, Y, Z$  of  $N$ . Two CGs are *Markov equivalent* if they represent the same independence model.

Let  $X, Y, Z$  and  $W$  denote four disjoint subsets of  $N$ . An independence model  $M$  is a *semi-graphoid* if it has the following properties: (i) *symmetry*:  $X \perp_M Y | Z$  implies  $Y \perp_M X | Z$ ; (ii) *decomposition*:  $X \perp_M Y \cup W | Z$  implies  $X \perp_M Y | Z$ ; (iii) *weak union*:  $X \perp_M Y \cup W | Z$  implies  $X \perp_M Y | Z \cup W$ ; and (iv) *contraction*:  $X \perp_M Y | Z \cup W$  and  $X \perp_M W | Z$  together imply  $X \perp_M Y \cup W | Z$ . A semi-graphoid  $M$  is a *graphoid* if it has the property (v) *intersection*:  $X \perp_M Y | Z \cup W$  and  $X \perp_M W | Z \cup Y$  together imply  $X \perp_M Y \cup W | Z$ . A graphoid  $M$  is *compositional* if  $X \perp_M Y | Z$  and  $X \perp_M W | Z$  together imply  $X \perp_M Y \cup W | Z$ . The independence model induced by a probability distribution is a semi-graphoid, and the independence model induced by a strictly positive probability distribution is a graphoid (Studený, 2005). While the independence model induced by a probability distribution is not a compositional graphoid in general, independence models induced by Gaussian probability distributions are compositional (Studený, 2005).

The *elementary statements* of a semi-graphoid  $M$  are those of the form  $x \perp_M y | Z$ , where  $x, y \in N$ . A semi-graphoid  $M$  is determined by its elementary statements, meaning that every statement in  $M$  follows from the elementary statements by repeatedly applying the semi-graphoid properties (Studený, 2005, Lemma 2.2). Thus one can equiv-

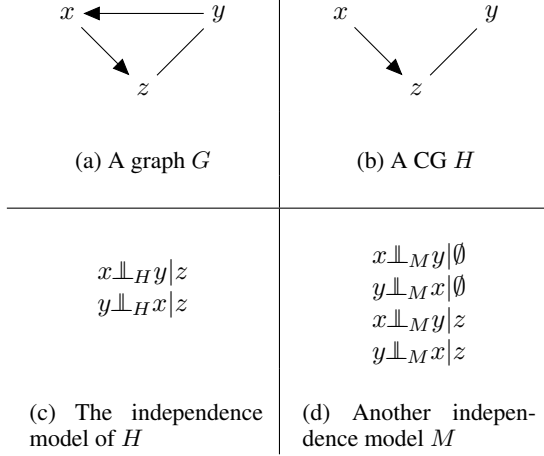


Figure 1: Example

alently work with the elementary statements of a semi-graphoid. We do so in the rest of the article. In other words, we restrict the independence model represented by a CG or induced by a probability distribution to its elementary statements.

A CG  $G$  is *inclusion optimal* wrt an independence model  $M$  if (i)  $I(G) \subseteq M$ , and (ii) there exists no CG  $H$  such that  $I(G) \subset I(H) \subseteq M$ . We say that an inclusion optimal  $G$  is *independence optimal* if there exists no CG  $H$  such that  $I(H) \subseteq M$  and  $|I(H)| > |I(G)|$ .

**Example 1** *Although the graph  $G$  shown in Figure 1a only contains directed and undirected edges, it is not a CG since it contains the semidirected cycle  $x \rightarrow z - y \rightarrow x$ . The graph  $H$  in Figure 1b is a CG; its independence model  $I(H)$  shown in Figure 1c. For example,  $x \perp_H y | z$  holds because the only route between  $x$  and  $y$  is  $x \rightarrow z - y$  and  $z$  is in the conditioning set. Moreover,  $I(H)$  fulfills the graphoid properties since it is symmetric and only contains one pair of independencies. Finally,  $I(H) \subset I(M)$  holds for the independence model  $M$  shown in Figure 1d. This means that  $H$  is inclusion optimal wrt  $M$ , since removing an edge from  $H$  would cause either  $x \perp_H z | \emptyset$  or  $y \perp_H z | \emptyset$  to be true, neither of which are in  $M$ . Also note that  $M$  is not faithful to any CG since there is no CG  $H'$  such that  $I(H') = I(M)$ .*

## 2.1 Algorithms for Learning CGs

We continue by a short overview of existing algorithms for CG learning. To our knowledge, only three such have been proposed earlier.

The first algorithm, proposed in (Studený, 1997), is a PC-like algorithm that first finds a skeleton, and then orients the edges according to a specific set of rules. The second algorithm, LCD, proposed by (Ma et al., 2008), uses a divide-and-conquer approach that allows for fast learning

of CGs especially in the space of sparse CGs. If the probability distribution at hand is faithful to a CG  $G$ , then both Studený's algorithm and LCD are guaranteed to find a CG  $H$  that is Markov equivalent with  $G$ .

The third algorithm for learning CGs, CKES, proposed in (Peña et al., 2014), can be viewed as an extension of the KES (Nielsen et al., 2003) Bayesian network learning algorithm. CKES works by iteratively adding (resp. removing) edges between the variables in the CG that are dependent (resp. independent) in the probability distribution at hand. The CKES algorithm guarantees to produce a CG that is inclusion optimal wrt the independence model induced by the probability distribution at hand in case the model is a compositional graphoid.

## 3 INFERENCE RULES FOR FINDING SEPARATIONS IN CHAIN GRAPHS

In this work, we rely on inference rules by Studený (1998) that allow for efficiently computing the separations and non-separations in a given CG  $G$ . We will represent these rules in our ASP encoding of CG learning, as detailed in Section 4. The rules are based on four sets of nodes  $U_x^C$ ,  $V_x^C$ ,  $W_x^C$ , and  $Z_x^C$ , that are saturated for each node  $x \in N$  wrt the conditioning set  $C \subseteq N \setminus \{x\}$ . A node  $x$  is then separated from each node  $y \in N \setminus (U_x^C \cup V_x^C)$  given the conditioning set  $C$ , i.e.,  $x \perp_G y | C$ . The rules are shown in Figure 2. For some intuition, note that

- $y \in V_x^C$  iff there exists a C-open route from  $x$  to  $y$  in  $G$  which contains the subroute  $a_i \rightarrow a_{i+1} - \dots - a_{i+k} = y$ ,  $k \geq 1$ ,
- $y \in U_x^C$  iff there exists a C-open route from  $x$  to  $y$  in  $G$  which does not contain the subroute  $a_i \rightarrow a_{i+1} - \dots - a_{i+k} = y$ ,  $k \geq 1$ ,
- $y \in W_x^C$  iff there exists a  $z \in U_x^C \cup V_x^C$  and a route  $z = a_0 \rightarrow a_1 - \dots - a_r = y$  in  $G$ ,  $r \geq 1$ , and
- $y \in Z_x^C$  iff there exists a  $z \in U_x^C \cup V_x^C$  and a route  $z = a_0 \rightarrow a_1 - \dots - a_r = y$ ,  $r \geq 1$ , with  $\{a_1, \dots, a_r\} \cap C \neq \emptyset$ .

Note that rule 7 in Figure 2 corrects a small typo in the original paper (Studený, 1998).

The correctness of these rules can be stated as follows.

### Theorem 1 (Adapted from (Studený, 1998))

*Given a CG  $G$  over a node set  $N$ , starting with  $U_x^C = V_x^C = Z_x^C = W_x^C = \emptyset$  for each node  $x \in N$  and  $C \subseteq N$ , apply the rules in Figure 2 until fixpoint. Then  $y \notin U_x^C \cup V_x^C$  iff  $x \perp_G y | C$ .*

|  |
|--|
| 0. $C \subset N, x \notin C \Rightarrow x \in U_x^C$                             |
| 1. $x \in U_a^C, x - y, y \notin C \Rightarrow y \in U_a^C$                      |
| 2. $x \in U_a^C, y \rightarrow x, y \notin C \Rightarrow y \in U_a^C$            |
| 3. $x \in U_a^C \cup V_a^C, x \rightarrow y, y \notin C \Rightarrow y \in V_a^C$ |
| 4. $x \in V_a^C, x - y, y \notin C \Rightarrow y \in V_a^C$                      |
| 5. $x \in U_a^C \cup V_a^C, x \rightarrow y \Rightarrow y \in W_a^C$             |
| 6. $x \in W_a^C, x - y \Rightarrow y \in W_a^C$                                  |
| 7. $x \in W_a^C, x \in C \Rightarrow x \in Z_a^C$                                |
| 8. $x \in Z_a^C, x - y \Rightarrow y \in Z_a^C$                                  |
| 9. $x \in Z_a^C, y \rightarrow x, y \notin C \Rightarrow y \in U_a^C$            |

Figure 2: Studený's inference rules

### 3.1 Refining Studený's Rules

It turns out that the  $W$  sets in Studený's original rules are actually redundant. In detail, the  $W$  sets can be removed by replacing the original rules 5–7 by the rules 10–11 presented in Figure 3.

|  |
|--|
| 10. $x \in U_a^C \cup V_a^C, x \rightarrow y, y \in C \Rightarrow y \in Z_a^C$ |
| 11. $x \in V_a^C, x - y, y \in C \Rightarrow y \in Z_a^C$                      |

Figure 3: Replacement for Studený's rules 5–7

**Proposition 1** *Given a CG  $G$  over a node set  $N$ , starting with  $U_x^C = V_x^C = Z_x^C = W_x^C = \emptyset$  for each node  $x \in N$  and  $C \subseteq N$ , the following computations give the same  $U_x^C$  and  $V_x^C$  sets:*

- Apply the rules 0–9 in Figure 2 until fixpoint.
- Apply rules 0–4 and rules 8–9 in Figure 2 together with rules 10–11 in Figure 3 until fixpoint.

*Proof (sketch).* Without loss of generality, perform the original and the refined computation by running the following three steps repeatedly until fixpoint. Step 1: Run rules 0–4 until fixpoint. Step 2: Run rules 5–8 (original computation) or rules 8,10–11 (refined computation) until fixpoint. Step 3: Run rule 9 until fixpoint. In order to prove the lemma, it suffices to prove that  $Z_x^C$  is the same at the end of step 2 for both computations.

Consider arbitrary  $U_x^C$  and  $V_x^C$ . Recall that by using the original rules 0–9, a node  $y \in Z_x^C$  only if there exists a

node  $z \in U_x^C \cup V_x^C$  and a route  $z = a_0 \rightarrow a_1 - \dots - a_r = y, r \geq 1$ , with  $\{a_1, \dots, a_r\} \cap C \neq \emptyset$ . Let  $a_s$  denote the first node in the subroute  $a_1 - \dots - a_r$  that is in  $C$ . Observe that  $a_{s-1} \in U_x^C \cup V_x^C$  if  $s = 1$ , and  $a_{s-1} \in V_x^C$  otherwise. One can check that we have exactly  $a_s \in Z_x^C$  both by applying rules 10–11 in the refined computation and by applying rules 5–7 in the original computation. After this,  $y \in Z_x^C$  is obtained both in the original and in the refined computation by repeatedly applying rule 8.  $\square$

### 3.2 Additional Rules

On top of the “replacement” rules 10–11, we further identify additional rules that can be applied soundly together with Studený's original rules, i.e., without affecting the resulting  $U$  and  $V$  sets. While these rules, shown in Figure 4, are redundant, encoding the rules declaratively as part of the ASP encoding presented in this work improves the overall running times of our approach in practice.

**Proposition 2** *Given a CG  $G$  over a node set  $N$ , starting with  $U_x^C = V_x^C = Z_x^C = W_x^C = \emptyset$  for each node  $x \in N$  and  $C \subseteq N$ , the following computations give the same  $U_x^C$  and  $V_x^C$  sets:*

- Apply the rules 0–9 in Figure 2 until fixpoint.
- Apply the rules 0–9 in Figure 2 together with rules 12–17 in Figure 4 until fixpoint.

*Proof (sketch).* Consider rule 12. According to the head of the rule, there exists a C-open route from  $a$  to  $x$  which does not contain the subroute  $a_i \rightarrow a_{i+1} - \dots - a_{i+k} = x$  with  $k \geq 1$ . Then, preceding this route with the edge  $b \leftarrow a$  with  $b \notin C$  results in a C-open route from  $b$  to  $x$  which does not contain the subroute  $a_i \rightarrow a_{i+1} - \dots - a_{i+k} = x$  with  $k \geq 1$ . Then, the body of the rule holds. The correctness of the rest of the rules can be proven in the same way.  $\square$

|  |
|--|
| 12. $x \in U_a^C, a \rightarrow b, b \notin C \Rightarrow x \in U_b^C$ |
| 13. $x \in V_a^C, a \rightarrow b, b \notin C \Rightarrow x \in V_b^C$ |
| 14. $x \in Z_a^C, a \rightarrow b, b \notin C \Rightarrow x \in Z_b^C$ |
| 15. $x \in U_a^C, y \in U_x^C \Rightarrow y \in U_a^C$                 |
| 16. $x \in U_a^C, y \in V_x^C \Rightarrow y \in V_a^C$                 |
| 17. $x \in U_a^C, y \in Z_x^C \Rightarrow y \in Z_a^C$                 |

Figure 4: Additional sound rules

## 4 LEARNING CHAIN GRAPHS VIA ASP

In this section we detail our method of learning CGs using ASP. The approach is constraint-based and allows for the objective function to take additional domain knowledge into account. We start with a short informal account of ASP.

### 4.1 Answer Set Programming

Answer set programming (ASP) is a rule-based declarative constraint satisfaction paradigm that is well-suited for representing and solving various computationally hard problems (Gelfond and Lifschitz, 1988; Niemelä, 1999; Simons et al., 2002). ASP offers an expressive declarative modelling language in terms of first-order logical rules, allowing for intuitive and compact representations of NP-hard optimization tasks. When using ASP, the first task is to model the problem in terms of ASP rules (constraints) so that the set of solutions implicitly represented by the ASP rules corresponds to the solutions of the original problem. One or multiple solutions of the original problem can then be obtained by invoking an off-the-shelf ASP solver on the constraint declaration. The algorithms underlying the ASP solver Clingo (Gebser et al., 2011) that we use in this work are based on state-of-the-art Boolean satisfiability solving techniques (Biere et al., 2009). These techniques have during the last 10-15 years emerged as robust and efficient means of solving various hard search and optimization problems and even in cases surpassed specialized algorithms while at the same time offering great flexibility as general NP-procedures for declarative problem solving. As a self-contained explanation of ASP syntax and semantics would exceed the page limit, we only aim to give an intuitive reading of our ASP encoding for learning CGs.

### 4.2 Encoding the CG structure learning problem

Our exact ASP encoding of the CG structure learning problem is modular, consisting of three parts: (1) a set of constraints representing the space of CGs with a given set of nodes, including constraints ruling out semidirected cycles (Section 4.2.1); (2) a set of rules exactly encoding the Studený inference rules and thereby the separations and non-separations of a given CG (Section 4.2.2); (3) a set of soft constraints exactly representing a well-defined objective function used for finding the optimal CG structure of a problem (Section 4.2.3). Note that parts 2 and 3 are dependent on the input data. Specifically, part 2 represents the dependencies and independencies that are determined by the data. Additional information about the dependencies and independencies can also be added to part 2, such as the confidence of their correctness. This information can then be used by the objective function in part 3 as discussed in Section 4.2.3. Essentially, by calling an ASP solver with the whole ASP encoding, consisting of parts 1–3, the solver

will perform an intelligent implicit search over the space of CGs (using part 1), and will output a CG that produces the best objective function score (based on part 3) by deriving the separations and non-separations in the CGs (using part 2). Furthermore, in Section 4.3 we discuss how additional domain knowledge can be incorporated.

#### 4.2.1 Encoding CGs

We start with part 1, i.e., by describing an ASP program encoding of the space of CGs. This base encoding is represented in Figure 5.

In the encoding, the input predicate *node* represents the fact that  $x$  is a node. We use the predicates *edge* and *arc* to represent the undirected and directed edges, respectively, of the CG  $G$ , i.e., these predicates represent the actual CG described by a solution to the ASP program. More precisely, we have that  $edge(X, Y)$  is true iff  $x - y \in G$ , and  $arc(X, Y)$  is true iff  $x \rightarrow y \in G$ . Informally, the “guess” part of the program, consisting of the two first rules, encodes a non-deterministic guess of the edges in  $G$ , which means that the ASP solver will implicitly consider all possible graphs during search. The next three rules enforce the fact that, for any pair  $x, y$  of nodes in a CG  $G$ , there can be at most one type of an edge (undirected, directed, or neither) between them. The predicate *ancestor*( $X, Y$ ) is true iff there is a semidirected route from a node  $x$  to node  $y$ , and is used to enforce that CGs cannot contain semidirected cycles. This constraint is enforced transitively by the last five rules in Figure 5.

```
%%% guess graph:
% directed edges
{ arc(X,Y) } :- node(X;Y), X!=Y,
               not edge(X,Y),
               not arc(Y,X).

% undirected edges
{ edge(X,Y) } :- node(X;Y), X!=Y,
                not arc(X,Y),
                not arc(Y,X).

%%% at most one edge between node pairs:
% symmetric undirected edge relation
edge(Y,X) :- edge(X,Y).
% disallow cases with both directed and
% undirected edges over a pair of nodes
:- edge(X,Y), arc(X,Y).
% disallow pairwise opposite arcs
:- arc(X,Y), arc(Y,X).

%%% disallow partially directed cycles
ancestor(X,X) :- node(X).
ancestor(X,Y) :- arc(X,Y).
ancestor(X,Y) :- edge(X,Y).
ancestor(X,Y) :- ancestor(X,Z),
                ancestor(Z,Y).
:- ancestor(X,Y), arc(Y,X).
```

Figure 5: ASP encoding of chain graphs

## 4.2.2 Encoding Studeny’s Rules

For inferring the separations that hold in a given CG  $G$ , we encode Studeny’s inference rules—or more precisely, the modified rule set consisting of the rules 0–4, 8–9, and 10–11 (recall Section 3)—in ASP. As shown in Figure 6, the ASP modelling language allows for very natural encoding of these rules. The predicates  $inU(X, A, C)$ ,  $inV(X, A, C)$ , and  $inZ(X, A, C)$ , respectively, are used for representing the facts that a node  $x \in U_a^C$ ,  $x \in V_a^C$ , and  $x \in Z_a^C$ , respectively. The auxiliary predicate  $in(C, X)$ , although its defining rule may seem somewhat complicated, simply represents the fact that node  $x \in C \subseteq N$ . As a technical detail, we use an index-representation for the conditioning sets and the nodes: the index of a given set  $C \subseteq N = \{1..|N|\}$  is represented as a binary vector  $b_{|N|} \dots b_2 b_1$ , where  $b_i = 1$  iff node  $i \in C$ . The sets  $C \subseteq N$  are represented by the input predicate  $set$ . Finally, the predicate  $derived\_dep(X, A, C)$  is defined to be true iff  $x \not\perp_G a | C$ , where  $G$  is the graph represented by the predi-

cates  $edge$  and  $arc$ , exactly following the conditions stated in Theorem 1.

## 4.2.3 Optimization

We proceed by detailing how different objective functions can be encoded within our approach. As noted earlier, the goal of an objective function is to assign a score to every CG  $G$ , and the set of optimal CGs is defined as those CGs which minimizes the objective function value. We consider three different objective functions that characterize different types of optimality conditions for CGs. Following Hyttinen et al. (2014)—who focused on causal structure discovery—each of the objective functions is (in some cases partially) based on how well the separations and non-separations of the given CG  $G$  corresponds to the independencies and dependencies determined from the data. In the following, we denote by  $CG(N)$  the set of CGs over the set of nodes  $N$ , and by  $I$  the (complete) set of independencies and dependencies given as input (determined before-hand from the data).

The first two objective functions follow the general idea of minimizing a sum of costs over incorrect separations and non-separations in  $G$  wrt the independencies and dependencies determined by the data. As a generalization, we associate for each (in)dependence statement determined by the data a cost (weight)  $w(x \perp y | C)$  and  $w(x \not\perp y | C)$ , which is incurred on the solution  $G$  iff the corresponding separation statement does not hold in  $G$ . This allows for formalizing a general weighted objective function

$$\min_{G \in CG(N)} \sum_{(x \not\perp y | C) \in I} w(x \not\perp y | C) \cdot T[x \perp y | C] + \sum_{(x \perp y | C) \in I} w(x \perp y | C) \cdot T[x \not\perp y | C], \quad (1)$$

where  $T[c]$  is an indicator for the condition  $c$  being true for graph  $G$ .

By setting the weights  $w$  appropriately, this gives various interesting special cases. For example, to represent independence optimal CGs as the optimal solutions, let  $w(x \not\perp y | C) = \infty$  (i.e., all dependencies have to be represented in any CG  $G$ ) and  $w(x \perp y | C) = 1$  (i.e., each independence relation not represented by  $G$  adds a unit cost to  $G$ ).

On the level of our ASP encodings of the objective functions considered, we use the input predicates  $indep(X, Y, C, W)$  and  $dep(X, Y, C, W)$ , to represent  $w(x \perp y | C)$  and  $w(x \not\perp y | C)$ , respectively. Using these predicates, the general weighted objective function is expressed as the ASP statements shown in Figure 7.

In addition to allowing for associating the same (unit) weight to all (in)dependence statements, the general weighted objective function also allows for using more elaborate weighting schemes where each (in)dependence statement have a different weight. In contrast, most

```

% in(C, X): node X in set C
in(C, X) :- set(C), node(X),
            2**(X-1) & C != 0.

% rule 0
inU(X, X, C) :- set(C), node(X),
               not in(C, X).

% rule 1
inU(Y, A, C) :- inU(X, A, C), edge(X, Y),
               not in(C, Y), not in(C, A).

% rule 2
inU(Y, A, C) :- inU(X, A, C), arc(Y, X),
               not in(C, Y), not in(C, A).

% rule 3
inV(Y, A, C) :- inU(X, A, C), arc(X, Y),
               not in(C, Y), not in(C, A).
inV(Y, A, C) :- inV(X, A, C), arc(X, Y),
               not in(C, Y), not in(C, A).

% rule 4
inV(Y, A, C) :- inV(X, A, C), edge(X, Y),
               not in(C, Y), not in(C, A).

% rule 8
inZ(Y, A, C) :- inZ(X, A, C), edge(X, Y),
               not in(C, A).

% rule 9
inU(Y, A, C) :- inZ(X, A, C), arc(Y, X),
               not in(C, Y), not in(C, A).

% rule 10
inZ(Y, A, C) :- inU(X, A, C), arc(X, Y),
               not in(C, A), in(C, Y).
inZ(Y, A, C) :- inV(X, A, C), arc(X, Y),
               not in(C, A), in(C, Y).

% rule 11
inZ(Y, A, C) :- inV(X, A, C), edge(X, Y),
               not in(C, A), in(C, Y).

% Derived connections
derived_dep(X, Y, C) :- inU(Y, X, C), X != Y,
                      not in(C, Y), not in(C, X).
derived_dep(X, Y, C) :- inV(Y, X, C), X != Y,
                      not in(C, Y), not in(C, X).

```

Figure 6: ASP encoding of Studeny’s rules

```

% Minimize the sum of the weights of
% unsatisfied (in)dependence constraints
:~indep(X,Y,C,W), derived_dep(X,Y,C).
                                     [W,X,Y,C]
:~dep(X,Y,C,W), not derived_dep(X,Y,C).
                                     [W,X,Y,C]

```

Figure 7: Encoding the general weighted objective

constraint-based structure learning methods only allow for binary information about an independence statement, which means that possible additional information about the independence statements, such as the level of confidence in them, is lost. In fact, as suggested by Hyttinen et al. (2014), by developing weighting schemes, one can bring the constraint-based approaches closer to score-based methods which have traditionally used local scores as weights within the implemented objective functions. For example, a weighting scheme that was proposed and shown to produce good solutions by Hyttinen et al. (2014) uses the log of the Bayesian probability of (in)dependence statements (Margaritis and Bromberg, 2009) as weights.

As an alternative to expressing optimality in terms of weighting incorrectly represented (in)dependencies, we also consider a novel objective function that aims at minimizing the number of edges in the CG while at the same time representing all conditional dependencies in the data. The underlying idea of the objective function is to minimize the model complexity and although independence optimal solutions can no longer be guaranteed wrt the independence model of the data, any solution will still be inclusion optimal. Formally, this gives the alternative objective function

$$\begin{aligned}
\min_{G \in \text{CG}(N)} \quad & |\{(x, y) : x < y, x - y \in G\}| + \\
& |\{(x, y) : x \rightarrow y \in G\}| + \\
& \sum_{(x \perp\!\!\!\perp y | C) \in I} w(x \perp\!\!\!\perp y | C) \cdot T[x \perp\!\!\!\perp_G y | C], \quad (2)
\end{aligned}$$

where  $w(x \perp\!\!\!\perp y | C) = \infty$ . This objective function is encoded in ASP as shown in Figure 8, which in itself demonstrates the versatility of our declarative approach.

### 4.3 Imposing Additional Constraints

One of the major strengths with our CG learning approach is the possibility to incorporate further constraints to guide the search for optimal solutions, in order to e.g. speed up the search for solution or to focus the search on specific solutions of interest. In terms of domain knowledge, if we know that variable  $x$  is the direct cause of variable  $y$  in the system we are modelling, we can simply add the constraint  $\text{arc}(x, y)$  as input to the ASP solver. In terms of including redundant constraints for speeding up search, one can e.g.

```

% satisfy all dependence constraints
:- dep(X,Y,C,_), not derived_dep(X,Y,C).

% minimize undirected and directed edges
:~ edge(X,Y), X<Y. [1,X,Y]
:~ arc(X,Y). [1,X,Y]

```

Figure 8: Encoding the minimize-edges objective

encode the set of redundant rules for how the sets  $U_x^C, V_x^C$  and  $Z_x^C$  can be computed as discussed in Section 3.2. An encoding of the rules are shown in Figure 9. In fact, we observed that the solver running times decreased by approximately 50% after adding these rules to the ASP encoding.

In terms of focusing search specific solutions of interest, we now describe an ASP encoding for restricting the search to only consider so-called *largest chain graphs* (Frydenberg, 1990; Volf and Studený, 1999). Largest CGs are representatives of Markov equivalence classes of CGs. Given a Markov equivalence class, the largest CG in the class is the CG which includes a maximal number of undirected edges. More formally, largest chain graphs are defined as follows.

A *complex* in a CG  $G$  is a path  $v_1, \dots, v_k$  in  $G$  with  $k \geq 3$ , such that  $v_1 \rightarrow v_2, v_i - v_{i+1}$  for  $i = 2, \dots, k-2$ , and  $v_{k-1} \leftarrow v_k$ , and there are no other edges between the nodes  $v_1, \dots, v_k$  in  $G$ . A *complex edge* is a directed edge that belongs to a complex. A directed edge  $u \rightarrow v$  covers a directed edge  $x \rightarrow y$  in a CG  $G$  if  $u$  is an ancestor of  $x$  and  $y$  is an ancestor of  $v$  in  $G$ . A directed edge  $u \rightarrow v$  is *protected* in  $G$  if it covers a complex edge. Note that every complex edge is protected. Now, a CG  $G$  is the largest CG in a Markov equivalence class iff every directed edge of  $G$  is protected.

An ASP encoding of the largest CG constraint is presented in Figure 10. Here, the predicate  $\text{complex}(X, Y)$  indicates that  $x \rightarrow y$  is a complex edge. The auxiliary predicate  $\text{almost\_undirected\_path}(X, Y, Z)$  indicates the existence of a path  $x \rightarrow y - \dots - z$  such that there are no other edges between the nodes in the path. This predicate

```

inU(Z,Y,C) :- inU(Z,X,C), arc(X,Y),
              not in(C,X), not in(C,Y), not in(C,Z).
inV(Z,Y,C) :- inV(Z,X,C), arc(X,Y),
              not in(C,X), not in(C,Y), not in(C,Z).
inZ(Z,Y,C) :- inZ(Z,X,C), arc(X,Y),
              not in(C,X), not in(C,Y).

inU(Y,A,C) :- inU(X,A,C), inU(Y,X,C).
inV(Y,A,C) :- inU(X,A,C), inV(Y,X,C).
inZ(Y,A,C) :- inU(X,A,C), inZ(Y,X,C).

```

Figure 9: Encoding redundant domain knowledge

```

% derive complex arcs X->Y - ... <-Z
complex(X,Y) :- arc(X,Y), arc(Z,Y), X!=Z,
               not edge(X,Z),
               not arc(X,Z), not arc(Z,X).
complex(X,Y) :- almost_undirected_path(X,Y,V),
               almost_undirected_path(Z,V,Y),
               X!=Z, not edge(X,Z),
               not arc(X,Z), not arc(Z,X).
almost_undirected_path(X,Y,Z) :- arc(X,Y),
                                 edge(Y,Z), not arc(X,Z).
almost_undirected_path(X,Y,Z) :-
  almost_undirected_path(X,Y,V),
  edge(V,Z), not arc(X,Z).

% arc U->V covers arc X->Y
covers(U,V,X,Y) :- ancestor(U,X),
                   ancestor(Y,V),
                   arc(X,Y), arc(U,V).

% all arcs must be protected
1 {covers(U,V,X,Y) : complex(X,Y)} :-arc(U,V).

```

Figure 10: ASP encoding of the largest CG constraint

is used for deriving the predicate *complex*. The predicate *covers*( $U, V, X, Y$ ) encodes that  $u \rightarrow v$  covers the edge  $x \rightarrow y$ . The final cardinality constraint states that every directed edge  $u \rightarrow v$  must cover at least one complex edge  $x \rightarrow y$ , i.e.,  $u \rightarrow v$  has to be protected.

## 5 EMPIRICAL EVALUATION

We report here on an empirical evaluation comparing our declarative ASP-based approach with existing CG structure learning algorithms in terms of accuracy of the produced solutions. For solving the ASP encodings, we used the state-of-the-art ASP solver Clingo version 4.4.0. The CG learning algorithms we compare to are LCD (Ma et al., 2008) and CKES (Peña et al., 2014), for which we obtained implementations from the respective authors. We excluded the PC-like algorithm (Studený, 1997) from the comparison since Ma et al. (2008) have shown that it is outperformed by LCD.

For the ASP approach, we illustrate its diversibility by applying three different objective functions.

- (i) ASP-Indep, which implements the general weighted optimization function with  $w(x \perp\!\!\!\perp y|C) = \infty$  and  $w(x \perp\!\!\!\perp y|C) = 1$  (cf. Section 4.2.3, Equation 1);
- (ii) ASP-Weight, which implements the general weighted optimization function using a probability-based weighting scheme from (Hytinen et al., 2014, Section 4.3), following a Bayesian paradigm to assign probabilities to (in)dependence statements, using code from the authors;
- (iii) ASP-Edge, which implements the minimize edges optimization function, (cf. Section 4.2.3, Equation 2).

Since the LCD algorithm assumes faithfulness, to some CG, from the probability distribution it is trying to learn, and does not work properly otherwise, we enforced this assumption for the experiments. However, we want to emphasize that our approach works also without such assumptions. Hence, due to this strong fairness towards LCD and CKES, the empirical results may to some extent present overly positive results for LCD and CKES.

Assuming a faithful underlying probability distribution, we applied the following process. First, a set of CGs  $G$  were generated with corresponding probability distributions. These probability distributions were then sampled into datasets, each determining a set of independence and dependence statements, forming the inputs to the different algorithms. We then evaluated the learning results, i.e., the CGs output as solutions by the algorithms, by comparing them to the original CG  $G$ .

For the experiments, we generated 100 “original” CGs over  $N = 7$  nodes, with an average node-degree of 2, as well as corresponding Gaussian probability distributions. For each distribution, we obtained 500 samples. We used the simplified Wilks independence test (Wilks, 1938) for LCD, CKES, ASP-Indep and ASP-Edge. For the ASP-Weight variant, we used the more advanced, non-binary independence test following (Hytinen et al., 2014). Taking into account that CKES typically ends up in different local optima (Peña et al., 2014), we ran the CKES algorithm three times on each input, and report the best found solution—wrt independence optimality to the independence model determined by the data—as the final solution produced by CKES.

To evaluate the accuracy of the learnt CGs we compared their represented independence model with the independence model of the original CG  $G$  by calculating the true positive rate (TPR) and false positive rate (FPR) of learnt dependencies. A higher TPR means that more dependencies are correctly identified by the algorithm, while a lower FPR means that more independencies are correctly identified.

The accuracy of the results is plotted in the ROC space in Figure 11.<sup>1</sup> We observe that LCD does not achieve as high TPR as the other algorithms, but does obtain relatively low FPR. This is in line with earlier results (Ma et al., 2008; Peña et al., 2014) that show that LCD includes edges relatively cautiously. CKES, in turn, achieves relatively high TPR, but with higher FPR, which is in line with previous evidence suggesting that CKES can have problems identifying the correct independencies in a probabil-

<sup>1</sup>Note that the points are not statistical test results obtained independently of each other. Rather, they are learning results of algorithms for which the dependencies and independencies logically constrain each other. As a result, some of the algorithms are not able to produce results with low FPRs at all, making the curves non-concave.

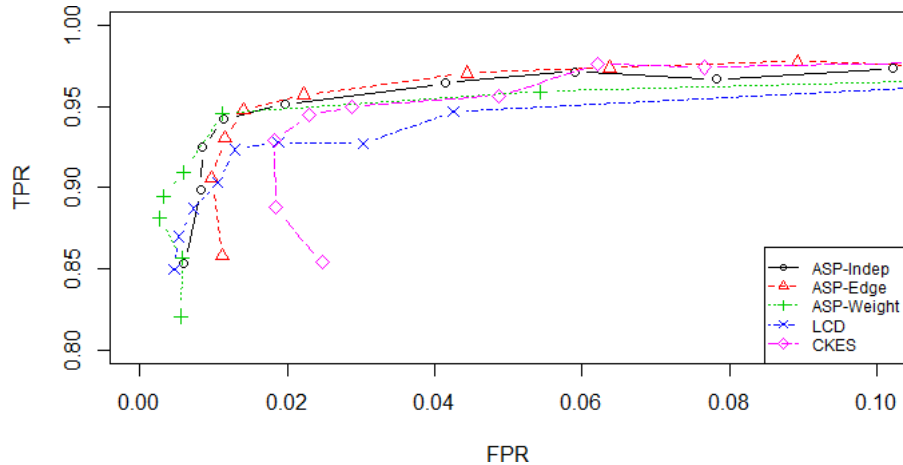


Figure 11: Comparison of solution accuracy

ity distribution and hence is relatively generous in including edges (Peña et al., 2014). In contrast, relative to LCD and CKES, we observe that the ASP-based approach under the different objective functions perform well both in the low FPR range (similarly or better than LCD which in turn dominates CKES in the range) and the higher FPR range (similarly or better than CKES which is in turn dominates LCD in the range). Thus the overall best-performing approaches are the ASP encoding variants. Among the ASP encoding variants, we observe that the ASP-Weight variant shows best performance in the low FPR range  $< 0.01$ . The variants ASP-Weight and ASP-Indep, using instantiations of the general weighted objective function, fare better than ASP-Edge that optimally minimizes the number of edges in the output CG.

As for scalability of our approach, the average and median running times over 100 inputs in terms of the number of variables are shown in Table 1, using a single core of an Intel i5 processor running at 3.4 GHz. For a fair comparison, we note that CKES and LCD do exhibit much faster running times, e.g., 1-10 seconds per run at  $N = 7$ . Note that the median ASP running times are much lower than the average running times. We observed that relatively scarce outliers have a big negative influence on the averages. The running times include both the time it takes to reach an optimal solution as well as the additional time it takes for the solver to *prove* that the found solution is optimal. In fact,

typically proving optimality consumes the majority of the running time of the solver. Moreover, during the search, the solver actually reports increasingly good solutions, constituting an anytime learning algorithm.

## 6 CONCLUSIONS

In this article we presented a first exact approach to chain graph structure learning. In the approach the computationally hard problem is cast as a constraint optimization problem using the declarative programming of answer set programming. In contrast to previously proposed CG learning algorithms, our approach enables finding provably optimal solutions to the learning problem without making assumptions on the data. It is at the same time more general since it enables the use of various types of objective functions for optimization and specifying domain knowledge in terms of hard constraints. Via an empirical evaluation we also showed that the approach exhibits better overall accuracy than any single previously proposed CG structure learning algorithm. A topic for further work is to study ways of scaling up the approach to higher numbers of variables while still maintaining optimality guarantees. It would also be interesting to extend the approach to accommodate the alternative multivariate regression (Cox and Wermuth, 1993, 1996) and Andersson-Madigan-Perlman (Andersson et al., 2001; Levitz et al., 2001) CG interpretations.

Table 1: ASP running times under different objectives

| $N$ | Median / Average (sec) |                 |                 |
|-----|------------------------|-----------------|-----------------|
|     | ASP-Indep              | ASP-Edge        | ASP-Weight      |
| 3   | 0.02 / 0.02            | 0.02 / 0.02     | 0.02 / 0.02     |
| 4   | 0.03 / 0.03            | 0.04 / 0.04     | 0.03 / 0.03     |
| 5   | 0.11 / 0.15            | 0.33 / 0.35     | 0.24 / 0.46     |
| 6   | 0.93 / 2.06            | 7.84 / 10.10    | 5.00 / 270.35   |
| 7   | 21.35 / 243.10         | 366.86 / 489.71 | 60.70 / 2471.81 |

**Acknowledgements** This work was supported in part by the Swedish Research Council (2010-4808), Academy of Finland (grants 251170 COIN Centre of Excellence in Computational Inference Research, 276412, and 284591), and Research Funds of the University of Helsinki.



## References

- Andersson, S., Madigan, D., and Perlman, M. (2001). An alternative Markov property for chain graphs. *Scandinavian Journal of Statistics*, 28:33–85.
- Berg, J., Järvisalo, M., and Malone, B. (2014). Learning optimal bounded treewidth Bayesian networks via maximum satisfiability. In *Proc. AISTATS*, volume 33 of *JMLR Proceedings*, pages 86–95. JMLR.org.
- Biere, A., Heule, M., van Maaren, H., and Walsh, T., editors (2009). *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*. IOS Press.
- Corander, J., Janhunen, T., Rintanen, J., Nyman, H. J., and Pensar, J. (2013). Learning chordal markov networks by constraint satisfaction. In *Proc. NIPS*, pages 1349–1357.
- Cox, D. and Wermuth, N. (1993). Linear dependencies represented by chain graphs. *Statistical Science*, 8:204–218.
- Cox, D. and Wermuth, N. (1996). *Multivariate Dependencies: Models, Analysis and Interpretation*. Chapman and Hall.
- Cussens, J. and Bartlett, M. (2013). Advances in Bayesian network learning using integer programming. In *Proc. UAI*, pages 182–191. AUAI Press.
- Frydenberg, M. (1990). The chain graph Markov property. *Scandinavian Journal of Statistics*, 17:333–353.
- Gebser, M., Kaufmann, B., Kaminski, R., Ostrowski, M., Schaub, T., and Schneider, M. T. (2011). Potassco: The Potsdam answer set solving collection. *AI Communications*, 24(2):107–124.
- Gelfond, M. and Lifschitz, V. (1988). The stable model semantics for logic programming. In *Proc. ICLP*, pages 1070–1080.
- Hytinen, A., Eberhardt, F., and Järvisalo, M. (2014). Constraint-based causal discovery: Conflict resolution with answer set programming. In *Proc. UAI*, pages 340–349. AUAI Press.
- Hytinen, A., Hoyer, P. O., Eberhardt, F., and Järvisalo, M. (2013). Discovering cyclic causal models with latent variables: A general SAT-based procedure. In *Proc. UAI*, pages 301–310. AUAI Press.
- Lauritzen, S. and Wermuth, N. (1989). Graphical models for association between variables, some of which are qualitative and some quantitative. *The Annals of Statistics*, 17:31–57.
- Levitz, M., Perlman, M., and Madigan, D. (2001). Separation and completeness properties for AMP chain graph Markov models. *The Annals of Statistics*, 29:1751–1784.
- Ma, Z., Xie, X., and Geng, Z. (2008). Structural learning of chain graphs via decomposition. *Journal of Machine Learning Research*, 9:2847–2880.
- Margaritis, D. and Bromberg, F. (2009). Efficient Markov network discovery using particle filters. *Computational Intelligence*, 25(4):367–394.
- Nielsen, J., Kočka, T., and Peña, J. (2003). On local optima in learning Bayesian networks. In *Proc. UAI*, pages 435–442. AUAI Press.
- Niemelä, I. (1999). Logic programs with stable model semantics as a constraint programming paradigm. *Annals of Mathematics and Artificial Intelligence*, 25(3-4):241–273.
- Parviainen, P., Farahani, H. S., and Lagergren, J. (2014). Learning bounded tree-width Bayesian networks using integer linear programming. In *Proc. AISTATS*, volume 33 of *JMLR Proceedings*, pages 751–759. JMLR.org.
- Peña, J. M., Sonntag, D., and Nielsen, J. (2014). An inclusion optimal algorithm for chain graph structure learning. In *Proc. AISTATS*, volume 33 of *JMLR Proceedings*, pages 778–786. JMLR.org.
- Simons, P., Niemelä, I., and Soinen, T. (2002). Extending and implementing the stable model semantics. *Artificial Intelligence*, 138(1-2):181–234.
- Sonntag, D., Peña, J., and Gómez-Olmedo, M. (2015). Approximate counting of graphical models via MCMC revisited. *International Journal of Intelligent Systems*, 30:384–420.
- Studený, M. (1997). On recovery algorithms for chain graphs. *International Journal of Approximate Reasoning*, 17:265–293.
- Studený, M. (1998). Bayesian networks from the point of view of chain graphs. In *Proc. UAI*, pages 496–503. Morgan Kaufmann.
- Studený, M. (2005). *Probabilistic Conditional Independence Structures*. Springer.
- Volf, M. and Studený, M. (1999). A graphical characterization of the largest chain graphs. *Int. J. Approx. Reasoning*, 20(3):209–236.
- Wilks, S. (1938). The large-sample distribution of the likelihood ratio for testing composite hypotheses. *The Annals of Mathematical Statistics*, 20:595–601.

---

# How matroids occur in the context of learning Bayesian network structure

---

Milan Studený

the Institute of Information Theory and Automation of the CAS,  
Pod Vodárenskou věží 4, Prague, 182 08, Czech Republic

## Abstract

It is shown that any connected matroid having a non-trivial cluster of BN variables as its ground set induces a facet-defining inequality for the polytope(s) used in the ILP approach to globally optimal BN structure learning. The result applies to well-known  $k$ -cluster inequalities, which play a crucial role in the ILP approach.

## 1 INTRODUCTION

The motivation for this theoretical paper is learning *Bayesian network* (BN) structure. Some hidden connection of the theory of *matroids* to a recent trend in optimal BN structure learning is revealed; specifically, matroids are related to the application of *integer linear programming* (ILP) methods in this area. To explain the motivation, in this introductory section, the latest developments in the ILP approach to learning BN structure are recalled. Matroid theory is also briefly mentioned and the structure of the rest of the paper is described.

### 1.1 ILP APPROACH TO LEARNING

The usual score-based approach to BN structure learning consists in maximizing a *scoring criterion*  $G \mapsto \mathcal{Q}(G, D)$ , where  $G$  is an acyclic directed graph and  $D$  the observed database (Neapolitan, 2004). The value  $\mathcal{Q}(G, D)$  says how much the BN structure defined by the graph  $G$  fits the database  $D$ . Nevertheless, some researchers are used to identify the BN structure with the respective equivalence class of graphs and prefer to talk about learning the class.

Since the classic heuristic greedy equivalence search (GES) algorithm (Chickering, 2002) is known not to guarantee to find a globally optimal BN structure, researches started to look for alternative methods. One of them was based on the idea of *dynamic programming* (Silander, Myllymäki, 2006), which, however, was limited in the number of BN

variables (= nodes of the graph) because of the exponential growth of memory demands.

This limitation has been overcome by the ILP approach based on *family-variable vector* representation of the graphs suggested independently in (Jaakkola, Sontag, Globerson, Meila, 2010) and (Cussens, 2010). An important technical step to overcome the limitation was the idea of the reduction of the search space developed by de Campos and Ji and published in a later journal paper (2011).

Jaakkola *et al.* (2010) introduced an important class of *cluster-based* inequalities approximating the respective family-variable polytope and came with a method of gradual adding these special constraints. Cussens (2010) first applied the family-variable vector representation in the restricted context of pedigree learning. However, his next paper (Cussens, 2011), which was inspired by (Jaakkola *et al.*, 2010), dealt with general BN structure learning and came with a standard *cutting plane approach* offering a more efficient way of adding the (cluster) inequality constraints, based on solving a special simple sub-ILP problem. Moreover, his experiments with general-purpose cutting planes, the so-called Gomory cuts, lead him to the idea to introduce a wider class of *k-cluster inequalities*, where  $k$  is a natural number less than the cardinality of the cluster. Bartlett and Cussens (2013) extended later the cutting plane method to a more general *branch-and-cut* approach; they included a lot of fine improvements and achieved much better running times. One of the morals of their paper was that using additional facet-defining inequalities for the family-variable polytope can speed up the computation.

An alternative ILP approach based on *characteristic-imset* representation of BN structures appeared in (Hemmecke, Lindner, Studený, 2012); its motivational sources date back to the method of imsets from (Studený, 2005). Unlike the family-variable vectors, the characteristic imsets uniquely correspond to BN structures. However, although this ILP approach is also feasible, the computational experiments reported in (Studený, Haws, 2014) have not resulted in better running times in comparison with the 2013-year version of GOBNILP software (Cussens and Bartlett, 2015).

Our recent manuscript (Cussens, Haws, Studený, 2015) has been devoted to the comparison of the facet-defining inequalities for the family-variable polytope and for the characteristic-imset polytope. Note that the facet-defining inequalities appear to be the most useful ones in the cutting plane approach to solving ILP problems; see the reasons in §9.1-9.2 of (Wolsey, 1998). In (Cussens *et al.*, 2015), we established a one-to-one correspondence between *extreme supermodular set functions* and certain facet-defining inequalities for both polytopes. An important special case of such facet-defining inequalities are the above mentioned  $k$ -cluster constraints, which can be transformed to the characteristic-imset context.

## 1.2 MATROID THEORY

The theory of matroids had been formed in the 1930's as an abstract theory of independence and since then became one of important topics in combinatorial optimization. The reader is referred to Oxley's book (1992) for numerous examples of how matroids pervade various branches of discrete mathematics and for how they appear to be useful in computer science.

In (Cussens *et al.*, 2015) we observed an interesting relation of the above mentioned  $k$ -cluster inequalities to the so-called *connected uniform matroids*, which gives an elegant interpretation to those inequalities.

In this paper, I extend our former observation and, using an old result by Nguyen (1978) from matroid theory, show that any *connected matroid* over a cluster of BN variables involving at least two variables induces a facet-defining inequality both for the family-variable polytope and for the characteristic-imset polytope.

In my opinion, this theoretical result broadens the class of available facet-defining inequalities which can be used in the cutting plane approach to solving ILP problems arising in the optimal BN structure learning area.

## 1.3 PAPER STRUCTURE

In §2 formal definitions of basic concepts are given: from the area of BN structure learning, the theory of polytopes, and matroid theory. The next §3 then recalls a few observations on the cone of supermodular set functions and the results from (Cussens *et al.*, 2015), (Nguyen, 1978) that are needed. These allows one to formulate and prove the main result in §4. An illustrating example is given in §5. The conclusions and the discussion are in §6.

## 2 BASIC CONCEPTS

Let  $N$  be a finite set of *BN variables*; to avoid a trivial case, assume  $n := |N| \geq 2$ . In statistical context, the elements of  $N$  correspond to random variables; in graphical context,

they correspond to nodes of (acyclic directed) graphs. Its subsets  $C \subseteq N$  with  $|C| \geq 2$ , called *clusters* in this paper, will serve as ground sets of the matroids discussed here.

## 2.1 STRUCTURE LEARNING CONCEPTS

The symbol  $\text{DAGs}(N)$  will denote the collection of all acyclic directed graphs over  $N$ , which means the graphs having  $N$  as the set of nodes. Given  $G \in \text{DAGs}(N)$  and  $a \in N$ , the symbol  $\text{pa}_G(a) := \{b \in N : b \rightarrow a \text{ in } G\}$ , is the *parent set* of the node  $a$ . A well-known equivalent definition of acyclicity of a directed graph  $G$  over  $N$  is the existence of a total order  $a_1, \dots, a_n$  of nodes in  $N$  such that, for every  $i = 1, \dots, n$ ,  $\text{pa}_G(a_i) \subseteq \{a_1, \dots, a_{i-1}\}$ ; we say then the order and the graph are *consonant*.

A *BN model* is a pair  $(G, P)$ , where  $G \in \text{DAGs}(N)$  and  $P$  a probability distribution on the joint sample space  $X_N := \prod_{a \in N} X_a$ , the Cartesian product of individual non-empty finite sample spaces  $X_a$  for variables  $a \in N$ , which factorizes according to  $G$ . An equivalent characterization of the factorization property is that  $P$  is *Markovian* with respect to  $G$ , which means it satisfies the conditional independence restrictions determined by  $G$  (Lauritzen, 1996). The *BN structure* defined by  $G$  is formally the class of Markovian probability distributions with respect to  $G$ .

Different graphs over  $N$  could be *Markov equivalent*, which means they define the same BN structure. The classic graphical characterization of equivalent graphs by Verma and Pearl (1991) states that two graphs are Markov equivalent if and only if they have the same adjacencies and immoralities. Recall that an *immorality* in  $G \in \text{DAGs}(N)$  is an induced subgraph of  $G$  of the form  $a \rightarrow c \leftarrow b$ , where the nodes  $a$  and  $b$  are not adjacent in  $G$ . Markov equivalence of  $G, H \in \text{DAGs}(N)$  will be denoted by  $G \sim H$ .

The task of *learning* the BN structure is to determine it on the basis of an observed (complete) *database*  $D$ , which is a sequence  $x_1, \dots, x_m$ ,  $m \geq 1$  of elements of the joint sample space  $X_N$ . This is often done by maximizing some *quality criterion*, also called a *score* or a *scoring criterion*, which is a bivariate real function  $(G, D) \mapsto \mathcal{Q}(G, D)$ , where  $G \in \text{DAGs}(N)$  and  $D$  a database. Examples of such criteria are Schwarz's (1978) Bayesian information criterion (BIC) and Bayesian Dirichlet equivalence (BDE) score (Heckerman, Geiger, Chickering, 1995). The reader is referred to (Neapolitan, 2004) for the definition of a relevant concept of *statistical consistency*.

A crucial technical assumption from the computational point of view (Chickering, 2002) is that  $\mathcal{Q}$  should be additively *decomposable*, which means, it has the form

$$\mathcal{Q}(G, D) = \sum_{a \in N} q_D(a | \text{pa}_G(a)), \quad (1)$$

where the summands  $q_D(* | *)$  are called *local scores*. All criteria used in practice satisfy this requirement.

Given an observed database  $D$ , the goal is to maximize  $G \mapsto \mathcal{Q}(G, D)$ . Since the aim is learn the BN structure, a natural assumption is that the criterion  $\mathcal{Q}$  to be maximized is *score equivalent* (Bouckaert, 1995), which means, for every database  $D$  and  $G, H \in \text{DAGs}(N)$ ,

$$\mathcal{Q}(G, D) = \mathcal{Q}(H, D) \quad \text{whenever } G \sim H.$$

Most of the criteria used in practice satisfy that.

## 2.2 POLYTOPES FOR LEARNING

We recall a few basic concepts from polyhedral geometry; see (Barvinok, 2002) or (Ziegler, 1995) for more details.

Below we deal with the Euclidean real vector spaces  $\mathbb{R}^\Gamma$ , where  $\Gamma \neq \emptyset$  is a non-empty finite index set. Given two vectors  $v, w \in \mathbb{R}^\Gamma$ , their scalar product will be denoted by

$$\langle v, w \rangle_\Gamma := \sum_{i \in \Gamma} v_i \cdot w_i,$$

or just by  $\langle v, w \rangle$  if there is no danger of confusion.

A *polytope*  $P$  is the convex hull of finitely many vectors from  $\mathbb{R}^\Gamma$ ; we only consider non-empty  $P$ . The *dimension* of  $P$ , denoted by  $\dim(P)$ , is the dimension of its affine hull, which is nothing but a translate of a linear subspace. The maximal number of affinely independent vectors in  $P$  is then  $\dim(P) + 1$ .

Given  $o \in \mathbb{R}^\Gamma$  and  $u \in \mathbb{R}$ , a linear inequality  $\langle o, v \rangle \leq u$  for  $v \in \mathbb{R}^\Gamma$  is called *valid* for  $P$  if it holds for any  $v \in P$ . The inequality is then called *tight* for a vector  $w \in P$  if  $\langle o, w \rangle = u$ . Given such valid linear inequality for  $P$  the corresponding *face* of  $P$  is its subset  $F \subseteq P$  of the form

$$F = \{ v \in P : \langle o, v \rangle = u \}.$$

One usually deals with valid inequalities that are tight for at least one vector  $w \in P$  in which case  $F \neq \emptyset$ . Then we will name the respective inequality *face-defining*. The function  $v \in \mathbb{R}^\Gamma \mapsto \langle o, v \rangle$  is typically a linear objective to be maximized; with little abuse of terminology we will then call  $o \in \mathbb{R}^\Gamma$  an *objective*.

A *facet* of a polytope  $P$  is its face of the dimension  $\dim(P) - 1$ . The corresponding inequality will be then called *facet-defining*. Given a (non-empty) facet  $F \subseteq P$  of a full-dimensional polytope  $P$  in  $\mathbb{R}^\Gamma$ , its facet-defining inequality is unique up to a positive multiple (of both  $o \in \mathbb{R}^\Gamma$  and  $u \in \mathbb{R}$ ). A well-known fundamental result in polyhedral geometry is that every full-dimensional polytope  $P$  with non-empty facets is specified as the set of vectors  $v \in \mathbb{R}^\Gamma$  satisfying all facet-defining inequalities for  $P$ . Thus,  $P$  is a *bounded polyhedron* and the facet-defining inequalities provide its minimal description in terms of equalities.

### 2.2.1 Family-Variable Polytope

The index set for our family-variable vectors will be

$$\Upsilon := \{ (a | B) : a \in N \ \& \ \emptyset \neq B \subseteq N \setminus \{a\} \}.$$

Given  $G \in \text{DAGs}(N)$ , the symbol  $\eta_G$  will denote the *family-variable vector* encoding it:

$$\eta_G(a | B) = \begin{cases} 1 & \text{if } B = \text{pa}_G(a), \\ 0 & \text{otherwise,} \end{cases} \quad \text{for } (a | B) \in \Upsilon.$$

The *family-variable polytope* is defined as the convex hull of the collection of all such vectors:

$$F := \text{conv}(\{ \eta_G \in \mathbb{R}^\Upsilon : G \in \text{DAGs}(N) \}).$$

Clearly,  $\dim(F) = |\Upsilon| = n \cdot (2^{n-1} - 1)$ .

One can re-write (1) in terms of  $\eta_G$  in this way:

$$\mathcal{Q}(G, D) = \sum_{a \in N} \sum_{B \subseteq N \setminus \{a\}} q_D(a | B) \cdot \eta_G(a | B), \quad (2)$$

which allows one to interpret  $\mathcal{Q}$  as (the restriction of) a linear function of  $\eta_G$ . In particular, the maximization of  $\mathcal{Q}$  over  $\text{DAGs}(N)$  turns into the task to maximize a linear function with the objective  $o(a | B) = q_D(a | B)$  for  $(a | B) \in \Upsilon$  over the family-variable polytope  $F$ . In other words, the local scores become the components of the respective objective vector  $o \in \mathbb{R}^\Upsilon$ .

The assumption of score equivalence of  $\mathcal{Q}$  then implies the respective objective satisfies, for every  $G, H \in \text{DAGs}(N)$ ,

$$G \sim H \Rightarrow \langle o, \eta_G \rangle_\Upsilon = \langle o, \eta_H \rangle_\Upsilon. \quad (3)$$

Thus, if (3) holds for  $o \in \mathbb{R}^\Upsilon$  we will say that it is a *score equivalent objective*, abbreviated as *an SE objective*.

Given a cluster  $C \subseteq N$ ,  $|C| \geq 2$ , of BN variables and a natural number  $k = 1, \dots, |C| - 1$  the inequality

$$k \leq \sum_{a \in C} \sum_{B \subseteq N \setminus \{a\} : |B \cap C| < k} \eta_G(a | B)$$

is valid for any  $G \in \text{DAGs}(N)$ : as the induced subgraph  $G_C$  is acyclic, the first  $k$  nodes in a total order of nodes in  $C$  consonant with  $G_C$  have at most  $k - 1$  parents in  $C$ . In particular, the inequality is valid for any  $\eta \in F$  in place of  $\eta_G$  and one can transform it into a standardized form:

$$\sum_{a \in C} \sum_{B \subseteq N \setminus \{a\} : |B \cap C| \geq k} \eta(a | B) \leq |C| - k. \quad (4)$$

We will call (4) the *k-cluster inequality* for  $C$ ; its version for  $k = 1$  is simply the *cluster inequality* for  $C$ . Every  $k$ -cluster inequality is facet-defining for  $F$  and the objective defining (4) is SE; see (Cussens *et al.*, 2015, Corol 4).

*Example 1* Consider  $N = \{a, b, c, d\} = C$  and  $k = 2$ . Then (4) takes the following form:

$$\begin{aligned} & [\eta(a|bc) + \eta(a|bd) + \eta(a|cd) + \eta(a|bcd)] \\ & + [\eta(b|ac) + \eta(b|ad) + \eta(b|cd) + \eta(b|acd)] \quad (5) \\ & + [\eta(c|ab) + \eta(c|ad) + \eta(c|bd) + \eta(c|abd)] \\ & + [\eta(d|ab) + \eta(d|ac) + \eta(d|bc) + \eta(d|abc)] \leq 2. \end{aligned}$$

### 2.2.2 Characteristic-Imset Polytope

The *characteristic imset* of  $G \in \text{DAGs}(N)$ , introduced in (Hemmecke *et al.*, 2012) and denoted below by  $c_G$ , is an element of the vector space  $\mathbb{R}^\Lambda$  where

$$\Lambda := \{S \subseteq N : |S| \geq 2\}.$$

Recall from (Studený, Haws, 2013) that  $c_G$  is a many-to-one linear function of  $\eta_G$ , the transformation is  $\eta \mapsto c_\eta$ :

$$c_\eta(S) = \sum_{a \in S} \sum_{B: S \setminus \{a\} \subseteq B \subseteq N \setminus \{a\}} \eta(a|B) \quad (6)$$

for  $S \in \Lambda$ . Thus, given  $G \in \text{DAGs}(N)$ , (6) can serve as the definition of  $c_G$  in which one substitutes  $\eta = \eta_G$ . A fundamental observation is that, for  $G, H \in \text{DAGs}(N)$ ,  $G \sim H$  if and only if  $c_G = c_H$  (Hemmecke *et al.*, 2012). In particular, the characteristic imset is a unique representative of the corresponding BN structure.

The *characteristic-imset polytope* is defined as follows:

$$C := \text{conv}(\{c_G \in \mathbb{R}^\Lambda : G \in \text{DAGs}(N)\}).$$

One can show that  $\dim(C) = |\Lambda| = 2^n - n - 1$ . Of course,  $C$  is the image of  $F$  by the linear mapping (6).

A notable fact is that any valid inequality for  $C$  induces a valid inequality for  $F$ : if  $\langle z, c \rangle_\Lambda \leq u$ , where  $z \in \mathbb{R}^\Lambda$  and  $u \in \mathbb{R}$ , is a valid inequality for  $c \in C$ , substitute (6) into  $\langle z, c_\eta \rangle_\Lambda \leq u$  and re-arrange the terms after the components of  $\eta$ . Indeed, since the image of  $\eta_G$  by (6) is just  $c_G$ , one gets an inequality valid for any  $\eta_G$ ,  $G \in \text{DAGs}(N)$ . Moreover, the induced inequality for  $\eta \in F$  is given by an SE objective: if  $G \sim H$ , one has  $c_G = c_H$  and, therefore,  $\langle z, c_G \rangle_\Lambda = \langle z, c_H \rangle_\Lambda$ .

In fact, there is a one-to-one correspondence between the valid inequalities for  $C$  and the valid inequalities for  $F$  given by SE objectives (Cussens *et al.*, 2015). Thus, these special valid inequalities for  $F$  can also be viewed as the valid inequalities for  $C$ , that is, interpreted in the context of  $C$ . This concerns many facet-defining inequalities for  $F$ : the  $k$ -cluster inequality (4) takes the following form in the characteristic-imset context, see (Cussens *et al.*, 2015, § 9):

$$\sum_{S \subseteq C, |S| \geq k+1} z(S) \cdot c(S) \leq |C| - k,$$

where  $z(S) = (-1)^{|S|-k-1} \cdot \binom{|S|-2}{|S|-k-1}$  for any such  $S$ .

*Example 2* Consider  $N = \{a, b, c, d\}$ . Then the 2-cluster inequality for  $C = \{a, b, c, d\}$  takes the form

$$c(abc) + c(abd) + c(acd) + c(bcd) - 2 \cdot c(abcd) \leq 2.$$

Indeed, the substitution of (6) in it gives just (5).

### 2.3 CONCEPTS FROM MATROID THEORY

Let us recall some definitions and basic facts from matroid theory; see (Oxley, 1992, chapters 1,2,4) for more details.

A *matroid* is a pair  $(C, \mathcal{I})$  where  $C$  is a finite set, called its *ground set*, and  $\mathcal{I}$  a non-empty class of subsets of  $C$ , called the *independent sets* (of the matroid), which is closed under subsets:  $I \in \mathcal{I}$ ,  $J \subseteq I$  implies  $J \in \mathcal{I}$  and satisfies the *independence augmentation axiom*:

$$\begin{aligned} & \text{if } I, J \in \mathcal{I} \text{ and } |J| < |I| \\ & \text{then } a \in I \setminus J \text{ exists with } J \cup \{a\} \in \mathcal{I}. \end{aligned}$$

We will also say that the matroid is *on the set*  $C$ .

A number of equivalent descriptions of the matroid  $(C, \mathcal{I})$  exists. Any matroid can be described by the class  $\mathcal{B}$  of its *bases*, which are inclusion-maximal independent sets. The above independence augmentation axiom implies that the sets in  $\mathcal{B}$  have the same cardinality. The shared cardinality of bases of a matroid is called its *rank*. A well-known fact is that  $\mathcal{B} \subseteq \mathcal{P}(C)$  is the class of bases of a matroid on  $C$  iff it is a non-empty class of subsets of  $C$  satisfying the following *basis exchange axiom*:

$$\begin{aligned} & \text{if } I, J \in \mathcal{B} \text{ and } a \in I \setminus J \\ & \text{then } b \in J \setminus I \text{ exists with } (I \setminus \{a\}) \cup \{b\} \in \mathcal{B}. \end{aligned}$$

The class  $\mathcal{D}$  of *dependent sets* of  $(C, \mathcal{I})$  consists of those subsets of  $C$  that are not independent sets. The *circuits* of the matroid are the inclusion-minimal dependent sets. A class  $\mathcal{C} \subseteq \mathcal{P}(C)$  is the class of circuits of a matroid on  $C$  iff it is a class of non-empty inclusion-incomparable subsets of  $C$  satisfying the following *circuit elimination axiom*:

$$\begin{aligned} & \text{if } K, L \in \mathcal{C}, K \neq L \text{ and } a \in K \cap L \\ & \text{then } M \in \mathcal{C} \text{ exists with } M \subseteq (K \cup L) \setminus \{a\}. \end{aligned}$$

We will also use the description of the matroid  $(C, \mathcal{I})$  in terms of its *rank function*, which is a function  $r$  on  $\mathcal{P}(C)$  defined as follows:

$$r(J) = \max\{|I| : I \subseteq J \text{ \& } I \in \mathcal{I}\} \quad \text{for any } J \subseteq C.$$

The rank functions of matroids on  $C$  are characterized as integer-valued set functions  $r : \mathcal{P}(C) \rightarrow \mathbb{Z}$  satisfying the following three conditions:

- if  $I \subseteq C$  then  $0 \leq r(I) \leq |I|$ ,

- if  $J \subseteq I \subseteq C$  then  $r(J) \leq r(I)$ ,
- if  $I, J \subseteq C$  then  $r(I \cup J) + r(I \cap J) \leq r(I) + r(J)$ .

A set  $S \subseteq C$  is called a *separator* of a matroid  $(C, \mathcal{I})$  if

$$r(S) + r(C \setminus S) = r(C).$$

The matroid is called *connected* if it has no other separators except for the trivial ones  $S = \emptyset$  and  $S = C$ . Observe that if  $(C, \mathcal{I})$  is connected and  $|C| \geq 2$  then  $\bigcup \mathcal{B} = \bigcup \mathcal{I} = C$ , for otherwise  $\bigcup \mathcal{I}$  is a non-trivial separator or  $r \equiv 0$ . An equivalent definition of a connected matroid on  $C$  is that, for every pair  $a, b \in C$ ,  $a \neq b$ , a circuit  $D \in \mathcal{C}$  exists with  $a, b \in D$ , see (Oxley, 1992, Prop 4.1.4).

The *dual matroid* to a matroid over  $C$  having  $\mathcal{B} \subseteq \mathcal{P}(C)$  as its class of bases is the matroid on  $C$  having

$$\mathcal{B}^* := \{C \setminus B : B \in \mathcal{B}\}$$

as its class of bases. The formula for the rank function  $r^*$  of the dual matroid is as follows:

$$r^*(J) = |J| - r(C) + r(C \setminus J) \quad \text{for } J \subseteq C, \quad (7)$$

see (Oxley, 1992, Prop 2.1.9). Another well-known basic fact is that the dual matroid to a connected matroid is connected as well, see (Oxley, 1992, Corol 4.2.8).

*Example 3* Consider  $C = \{a, b, c, d\}$  and put

$$\mathcal{B} = \{ \{a, b\}, \{a, c\}, \{a, d\}, \{b, c\}, \{b, d\}, \{c, d\} \}.$$

Clearly,  $\mathcal{B}$  is the class of bases of a matroid on  $C$ . The independent sets in it are subsets of  $C$  of cardinality at most two. The circuits are subsets of  $C$  of cardinality 3. The rank function only depends on the cardinality:

$$r(J) = \min \{ |J|, 2 \} \quad \text{for } J \subseteq C.$$

The form of  $r$  implies that the only separators are  $S = \emptyset$  and  $S = C$ . In particular, the matroid is connected. The dual matroid is itself.

The above example is a special matroid in a certain sense: for any integer  $0 \leq k \leq |C|$  the *uniform matroid* on  $C$  of the rank  $k$  has the collection of subsets of  $C$  of the cardinality at most  $k$  as its class of independent sets.

In this paper, the attention is limited to matroids which have clusters of BN variables  $C \subseteq N$ ,  $|C| \geq 2$  as their ground sets. Any matroid  $(C, \mathcal{I})$  on such cluster  $C$  can be interpreted as a matroid on  $N$  because  $\mathcal{I} \subseteq \mathcal{P}(C)$  can be viewed as a class of subsets of  $N$ . This kind of *trivial extension* on  $N$  leads to the rank function  $\bar{r} : \mathcal{P}(N) \rightarrow \mathbb{Z}$  given by

$$\bar{r}(S) = r(C \cap S) \quad \text{for any } S \subseteq N.$$

### 3 SUPERMODULAR FUNCTIONS

In (Cussens *et al.*, 2015, §7) a one-to-one correspondence has been established between extreme supermodular set functions and certain important facets of  $F$ , respectively of  $C$ . The relevant concepts are recalled in this section.

A real function  $m : \mathcal{P}(N) \rightarrow \mathbb{R}$  on subsets of the set  $N$  of BN variables will be called *standardized* if  $m(S) = 0$  for  $S \subseteq N$ ,  $|S| \leq 1$ , and *supermodular* if

$$\forall U, V \subseteq N \quad m(U) + m(V) \leq m(U \cup V) + m(U \cap V).$$

Mirror images of supermodular functions are *submodular* functions, defined by the converse inequalities; recall from §2.3 that rank functions of matroids are submodular.

#### 3.1 EXTREME SUPERMODULAR FUNCTIONS

The collection of standardized supermodular functions on  $\mathcal{P}(N)$ , viewed as a set of vectors in  $\mathbb{R}^{\mathcal{P}(N)}$ , is a pointed polyhedral cone. Therefore, it has finitely many extreme rays, which makes the following definition meaningful.

A standardized supermodular function  $m : \mathcal{P}(N) \rightarrow \mathbb{R}$  will be called *extreme* if it generates an extreme ray of the standardized supermodular cone. Recall that a non-zero vector  $v$  in a cone generates its extreme ray if the only summands in (positive) convex combinations of vectors from the cone yielding  $v$  are non-negative multiples of  $v$ .

Theorems 1 and 2 from (Cussens *et al.*, 2015) together give the next observation.

**THEOREM 1** *An inequality  $\langle o, \eta \rangle_{\Upsilon} \leq u$  for  $\eta \in \mathbb{R}^{\Upsilon}$ , where  $o \in \mathbb{R}^{\Upsilon}$ ,  $u \in \mathbb{R}$ , is facet-defining for  $F$  and defined by an SE objective  $o$  iff there exists an extreme standardized supermodular function  $m$  on  $\mathcal{P}(N)$  such that  $o$  is given by*

$$o(a | B) = m(\{a\} \cup B) - m(B) \quad \text{for } (a | B) \in \Upsilon, \quad (8)$$

*and  $u$  is the shared value of  $\langle o, \eta_H \rangle_{\Upsilon}$  for full graphs  $H$  over  $N$ , that is, for such  $H \in \text{DAGs}(N)$  in which every pair of distinct nodes is adjacent.*

*Example 4* Consider  $N = \{a, b, c, d\}$  and the set function

$$m(S) = \begin{cases} 2 & \text{if } S = N, \\ 1 & \text{if } |S| = 3, \\ 0 & \text{otherwise,} \end{cases} \quad \text{for } S \subseteq N.$$

Clearly,  $m$  is a standardized supermodular function; finer arguments why  $m$  is extreme are given later (Example 5). The formula (8) leads to the inequality (5). By Theorem 1, (5) is facet-defining for the family-variable polytope  $F$ .

Moreover, Corollary 6 in (Cussens *et al.*, 2015) says what is the role of the inequalities from Theorem 1 in the characteristic-imset context; here we have in mind the correspondence of the inequalities mentioned in §2.2.2.

COROLLARY 1 *Facet-defining inequalities*  $\langle o, \eta \rangle_{\Upsilon} \leq u$  for  $\eta \in F$  with SE objectives correspond to facet-defining inequalities  $\langle z, c \rangle_{\Lambda} \leq u$  for  $c \in C$  tight for the 1-imset, which is the vector in  $\mathbb{R}^{\Lambda}$  whose all components are ones.

### 3.2 SUBMODULARITY AND RANK FUNCTIONS

Mirror images of supermodular functions are submodular ones, which play an important role in matroid theory. It follows from the facts mentioned in § 2.3 that every rank function of a matroid belongs to the cone of non-decreasing submodular functions  $r$  with  $r(\emptyset) = 0$ . This is a pointed polyhedral cone and has finitely many extreme rays.

Nguyen (1978) was interested in the question when the rank function of a matroid generates an extreme ray of that cone. The next fact follows from his Theorem 2.1.5.

THEOREM 2 *Let  $C$  be a non-empty finite set and  $(C, \mathcal{I})$  a matroid on it such that  $C = \bigcup \mathcal{I}$ . Then its rank function  $r$  generates an extreme ray of the cone of non-decreasing submodular functions on  $\mathcal{P}(C)$  satisfying  $r(\emptyset) = 0$  iff the corresponding matroid  $(C, \mathcal{I})$  is connected.*

## 4 MAIN RESULT

LEMMA 1 Given a connected matroid  $(C, \mathcal{I})$  on  $C \subseteq N$ ,  $|C| \geq 2$  with the rank function  $r : \mathcal{P}(C) \rightarrow \mathbb{Z}$ , the function

$$m(S) := |C \cap S| - r(C \cap S) \quad \text{for } S \subseteq N, \quad (9)$$

is extreme standardized supermodular function on  $\mathcal{P}(N)$ .

**Proof:** Let us denote by  $R[C]$ , for  $C \subseteq N$ , the cone of submodular functions  $r^*$  on  $\mathcal{P}(C)$  such that  $r^*(\emptyset) = 0$  and  $r^*(C) - r^*(C \setminus \{a\}) = 0$  for any  $a \in C$ . Any function  $r^*$  in  $R[C]$  is necessarily non-decreasing. The dual matroid to  $(C, \mathcal{I})$  is connected; by Theorem 2, its rank function  $r^*$  given by (7) generates an extreme ray of the non-decreasing submodular cone. Since  $(C, \mathcal{I})$  is connected,  $\bigcup \mathcal{I} = C$  says  $r(\{a\}) = 1$  for any  $a \in C$ . Moreover, the dual matroid to the dual matroid is again  $(C, \mathcal{I})$ , which gives

$$1 = r(\{a\}) = r^{**}(\{a\}) \stackrel{(7)}{=} 1 - r^*(C) + r^*(C \setminus \{a\})$$

for any  $a \in C$ ; hence,  $r^*$  belongs to the smaller cone  $R[C]$ . This easily implies that  $r^*$  generates an extreme ray of  $R[C]$ , which fact allows one to observe by a minor consideration that its trivial extension

$$\overline{r^*}(S) := r^*(C \cap S) \quad \text{for } S \subseteq N,$$

generates an extreme ray of  $R[N]$ . Finally, the formula

$$m(S) = \overline{r^*}(N) - \overline{r^*}(N \setminus S) \quad \text{for } S \subseteq N,$$

defines a one-to-one linear transformation of the cone  $R[N]$  onto the cone of standardized supermodular functions  $m$

on  $\mathcal{P}(N)$  (in fact, the transformation is self-inverse). In particular,  $\overline{r^*} \mapsto m$  maps generators of extreme rays to generators of extreme rays, implying that  $m$  is extreme in the respective cone. Thus, one can write for any  $S \subseteq N$ :

$$\begin{aligned} m(S) &= \overline{r^*}(N) - \overline{r^*}(N \setminus S) = r^*(C) - r^*(C \setminus S) \\ &\stackrel{(7)}{=} \{|C| - r(C)\} - \{|C \setminus S| - r(C) + r(C \cap S)\} \\ &= |C \cap S| - r(C \cap S), \end{aligned}$$

which gives (9).  $\square$

*Example 5* Consider  $N = \{a, b, c, d\} = C$  and take the uniform matroid on  $C$  of rank 2 from Example 3. It is a connected matroid and, by Lemma 1, it induces through (9) an extreme supermodular function  $m$  from Example 4.

THEOREM 3 *Given a connected matroid  $(C, \mathcal{I})$  on a cluster  $C \subseteq N$ ,  $|C| \geq 2$  of BN variables, the inequality*

$$\sum_{a \in C} \sum_{B \subseteq N \setminus \{a\}: \exists D \in \mathcal{C} \ a \in D \subseteq B \cup \{a\}} \eta(a|B) \leq |C| - k, \quad (10)$$

where  $k$  is the rank of  $(C, \mathcal{I})$  and  $\mathcal{C}$  the collection of its circuits, is a facet-defining inequality for  $F$ .

**Proof:** By Lemma 1, (9) gives an extreme standardized supermodular function; one can apply Theorem 1 then. The upper bound  $u$  in the respective facet-defining inequality  $\langle o, \eta \rangle_{\Upsilon} \leq u$  for  $\eta \in F$  is the shared value  $\langle o, \eta_H \rangle_{\Upsilon}$  for all graphs  $H \in \text{DAGs}(N)$ . Using (8) one gets  $u = m(N)$ , that is,  $u = m(N) \stackrel{(9)}{=} |C| - r(C) = |C| - k$ .

The formula for the objective coefficients  $o(a|B)$ , where  $a \in N$  and  $B \subseteq N \setminus \{a\}$  (possibly empty) is then

$$\begin{aligned} o(a|B) &\stackrel{(8)}{=} m(\{a\} \cup B) - m(B) \\ &\stackrel{(9)}{=} |C \cap \{a\}| - r(C \cap (\{a\} \cup B)) + r(C \cap B), \end{aligned}$$

implying  $o(a|B) = 0$  if  $a \in N \setminus C$ . In case  $a \in C$  one has

$$o(a|B) = 1 - r(C \cap (\{a\} \cup B)) + r(C \cap B) = o(a|C \cap B).$$

Therefore, in the rest of the proof, we assume  $a \in C$  and  $B \subseteq C \setminus \{a\}$ ; our goal is to verify

$$o(a|B) = \begin{cases} 1 & \exists D \in \mathcal{C} \text{ with } a \in D \ \& \ D \subseteq B \cup \{a\}, \\ 0 & \text{otherwise,} \end{cases}$$

which clearly gives (10). We come from the above formula

$$o(a|B) = 1 - r(\{a\} \cup B) + r(B). \quad (11)$$

Having fixed  $a \in C$ , the coefficient are monotone

$$E \subseteq B \subseteq C \setminus \{a\} \Rightarrow o(a|B) \geq o(a|E) \quad (12)$$

because of submodularity of  $r$ :

$$\begin{aligned} o(a|B) - o(a|E) &\stackrel{(11)}{=} r(B) + r(\{a\} \cup E) - r(E) - r(\{a\} \cup B) \geq 0. \end{aligned}$$

As  $(C, \mathcal{I})$  is connected one has  $r(\{a\}) = 1$  for any  $a \in C$ , which gives

$$o(a | \emptyset) \stackrel{(11)}{=} 1 - r(\{a\}) + r(\emptyset) = 1 - 1 + 0 = 0.$$

Since the dual matroid is also connected, one has

$$o(a | C \setminus \{a\}) \stackrel{(11)}{=} 1 - r(C) + r(C \setminus \{a\}) \stackrel{(7)}{=} r^*(\{a\}) = 1.$$

In particular, the objective coefficients are either zeros or ones. In case a circuit  $D \in \mathcal{C}$  exists with  $a \in D \subseteq B \cup \{a\}$ , it is enough to show  $o(a | D \setminus \{a\}) = 1$  and apply (12). Indeed, by the definition of a circuit,  $D \setminus \{a\} \in \mathcal{I}$  and  $r(D \setminus \{a\}) = |D| - 1$ . One cannot have  $r(D) = |D|$ , for otherwise  $D \in \mathcal{I}$  contradicts the assumption  $D \in \mathcal{C}$ . Thus,  $r(D) = |D| - 1$  and one has

$$o(a | D \setminus \{a\}) \stackrel{(11)}{=} 1 - r(D) + r(D \setminus \{a\}) = 1.$$

It remains to show that  $o(a | B) = 0$  in the complementary case that no such  $D \in \mathcal{C}$  exists for  $B$ . By the definition of the rank function  $r$ , a set  $J \subseteq B$  exists with  $J \in \mathcal{I}$  and  $|J| = r(B)$ . It is enough to show  $\{a\} \cup J \in \mathcal{I}$  because then  $r(\{a\} \cup B) = |J| + 1$  (use submodularity of  $r$ ) and

$$o(a | B) \stackrel{(11)}{=} 1 - r(\{a\} \cup B) + r(B) = 1 - (|J| + 1) + |J| = 0.$$

Thus, assume for a contradiction that  $\{a\} \cup J \in \mathcal{D}$  is a dependent set and, by the definition of circuits, find  $D \in \mathcal{C}$  with  $D \subseteq \{a\} \cup J$ . Necessarily  $a \in D$ , for otherwise a contradictory conclusion  $J \in \mathcal{D}$  is derived. This implies  $a \in D \subseteq \{a\} \cup J \subseteq \{a\} \cup B$  contradicting the assumption that no such circuit  $D \in \mathcal{C}$  exists for  $B$ .  $\square$

The observation that the  $k$ -cluster inequalities (4) are facet-defining for the family-variable polytope easily follows from Theorem 3. Indeed, any uniform matroid on  $C \subseteq N$ ,  $|C| \geq 2$  of the rank  $k$ ,  $1 \leq k \leq |C| - 1$  is connected. This fact is illustrated by the following simple example.

*Example 6* Consider  $N = \{a, b, c, d\}$ ,  $C = \{a, b, c\}$  and  $k = 1$ . The uniform matroid on  $C$  of rank 1 has the bases  $\{a\}$ ,  $\{b\}$  and  $\{c\}$ . Thus, the class of its circuits is

$$\mathcal{C} = \{ \{a, b\}, \{a, c\}, \{b, c\} \}.$$

Since every pair of BN variables in  $C$  is contained in a circuit, it is a connected matroid. To get the specific form of the inequality (10) in this case realize that  $a \in C$  is contained in two circuits  $D \in \mathcal{C}$ , namely in  $\{a, b\}$  and in  $\{a, c\}$ . Thus, one has in (10) those terms  $\eta(a | B)$  where  $B \subseteq N \setminus \{a\}$  and either  $b \in B \Leftrightarrow \{a, b\} \subseteq B \cup \{a\}$  or  $c \in B$ . Thus, (10) takes the form

$$\begin{aligned} & [\eta(a | b) + \eta(a | c) + \eta(a | bc) \\ & \quad + \eta(b | bd) + \eta(b | cd) + \eta(b | bcd)] \\ & + [\eta(b | a) + \eta(b | c) + \eta(b | ac) \\ & \quad + \eta(b | ad) + \eta(b | cd) + \eta(c | acd)] \\ & + [\eta(c | a) + \eta(c | b) + \eta(c | ab) \\ & \quad + \eta(c | ad) + \eta(c | bd) + \eta(c | abd)] \leq 2, \end{aligned}$$

which is just the cluster inequality (4) for  $C$  with  $k = 1$ . Theorem 3 claims it is a facet-defining inequality for  $F$ .

Another instance of a uniform matroid was mentioned in Example 3; in this case, the inequality (10) turns into (5) from Example 1. The next example goes beyond the scope of  $k$ -cluster inequalities and uniform matroids.

*Example 7* Consider  $C = \{a, b, c, d\} = N$  and put

$$\mathcal{B} = \{ \{a, b\}, \{a, c\}, \{a, d\}, \{b, c\}, \{b, d\} \}.$$

Clearly,  $\mathcal{B}$  is the class of bases of a matroid on  $C$  of the rank 2. The rank function has the form

$$r(J) = \begin{cases} 0 & \text{if } J = \emptyset, \\ 1 & \text{if } J = \{c, d\} \text{ or } |J| = 1, \\ 2 & \text{otherwise,} \end{cases} \quad \text{for } J \subseteq C,$$

while the class  $\mathcal{C}$  of its circuits is

$$\mathcal{C} = \{ \{a, b, c\}, \{a, b, d\}, \{c, d\} \}.$$

As every pair of elements in  $C$  is contained in a circuit, it is a connected matroid. Theorem 3 says that the inequality

$$\begin{aligned} & [\eta(a | bc) + \eta(a | bd) + \eta(a | bcd)] \\ & + [\eta(b | ac) + \eta(b | ad) + \eta(b | acd)] \\ & + [\eta(c | d) + \eta(c | ab) \\ & \quad + \eta(c | ad) + \eta(c | bd) + \eta(c | abd)] \\ & + [\eta(d | c) + \eta(d | ab) \\ & \quad + \eta(d | ac) + \eta(d | bc) + \eta(d | abc)] \leq 2. \end{aligned} \tag{13}$$

is facet-defining for  $F$ . An interesting observation is that the inequality (13) defines the so-called 4B-type facet found by Bartlett and Cussens (2013); see (13) in §6 of their paper where  $\{v_1, v_4\} = \{a, b\}$  and  $\{v_2, v_3\} = \{c, d\}$ .

**COROLLARY 2** *The inequality (10) from Theorem 3 has the following form in the characteristic-imset mode:*

$$\sum_{T \in \Lambda, T \subseteq C} z(T) \cdot c(T) \leq |C| - k \quad \text{for } c \in \mathbb{R}^\Lambda, \tag{14}$$

$$\text{where } z(T) = - \sum_{L \subseteq T} (-1)^{|T \setminus L|} \cdot r(L)$$

are determined by the rank function  $r$  of the matroid. The inequality (14) defines a facet of  $C$  containing the 1-imset.

**Proof:** This follows from Lemma 10 and the formula (20) in (Cussens *et al.*, 2015) saying that  $\langle o, \eta \rangle_\Upsilon = \langle z, c_\eta \rangle_\Lambda$  where the coefficients  $z(T)$  for  $T \in \Lambda$  are given by the Möbius transform of the corresponding standardized supermodular function  $m$ , that is, by

$$z(T) = \sum_{L \subseteq T} (-1)^{|T \setminus L|} \cdot m(L) \quad \text{for } T \in \Lambda.$$



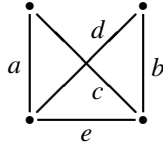


Figure 1: Edges of the graph define a matroid.

In our case,  $m$  is given by (9), which implies  $z(T) = 0$  whenever  $T \setminus C \neq \emptyset$ . Moreover, the Möbius transform of the first term in (9) vanishes for  $T \in \Lambda$ ,  $T \subseteq C$ , which gives (14). The second claim follows from Corollary 1.  $\square$

*Example 8* Consider again the matroid from Example 7. The formula (14) applied to the rank function  $r$  gives

$$z(T) = \begin{cases} -1 & \text{if } T = C, \\ 1 & \text{if } T \in \mathcal{C}, \\ 0 & \text{otherwise,} \end{cases} \quad \text{for } T \in \Lambda, T \subseteq C.$$

In particular, the inequality (13) has the following form in the characteristic-imset mode:

$$c(abc) + c(abd) + c(cd) - c(abcd) \leq 2.$$

## 5 FIVE VARIABLES EXAMPLE

Note that in case of four BN variables there is no other matroid-based facet-defining inequality for  $F$  except the  $k$ -cluster inequalities and (13). However, there are more matroid-based inequalities in case of five BN variables.

An important class of matroids are the so-called *graphic matroids* (Oxley, 1992, § 1.1). In fact, any undirected graph  $\mathcal{G}$  defines a matroid on *the set of its edges*. Specifically, a set  $I$  of edges in  $\mathcal{G}$  is considered to be *independent* (in the graphic matroid) if the edge-subgraph of  $\mathcal{G}$  consisting of edges in  $I$  is a *forest*, that is, has no undirected cycle.

The *circuits* of this graphic matroid are then the sets  $D$  of edges in  $\mathcal{G}$  forming edge-minimal cycles in  $\mathcal{G}$ , which means the removal of any edge from  $D$  results in a forest. The idea is illustrated by an example.

*Example 9* Consider  $C = \{a, b, c, d, e\} = N$  and define a matroid on  $C$  by means of the graph in Figure 1, where the edges are identified with the elements of  $C$ . It makes no problem to observe that the matroid has three circuits:

$$\mathcal{C} = \{ \{a, b, c, d\}, \{a, c, e\}, \{b, d, e\} \},$$

while the number of bases is eight: these are all 3-element subsets of  $C$  except for  $\{a, c, e\}$  and  $\{b, d, e\}$ . Of course, these are just the sets of edges defining spanning trees for the graph from Figure 1. It is easy to see that the matroid is connected and has rank  $k = 3$ . Like in Example 6 one

can determine the terms  $\eta(* | B)$  which occur in (10). For example,  $a \in C = N$  is contained in two circuits  $D \in \mathcal{C}$ , namely in  $\{a, c, e\}$  and  $\{a, b, c, d\}$ . In particular, one has in (10) those terms  $\eta(a | B)$  where  $B \subseteq N \setminus \{a\}$  and either  $\{c, e\} \subseteq B$  or  $\{b, c, d\} \subseteq B$ . The same principle applies to  $b, c, d$  and  $e$  which results in the following abbreviated form of (10):

$$\begin{aligned} & \sum_{ce \subseteq B \vee bcd \subseteq B} \eta(a | B) + \sum_{de \subseteq B \vee acd \subseteq B} \eta(b | B) \\ & + \sum_{ae \subseteq B \vee abd \subseteq B} \eta(c | B) + \sum_{be \subseteq B \vee abc \subseteq B} \eta(d | B) \\ & + \sum_{ac \subseteq B \vee bd \subseteq B} \eta(e | B) \leq 2. \end{aligned} \quad (15)$$

Thus, by Theorem 3, the inequality (15) is facet-defining for  $F$ . We leave to the reader to derive the rank function  $r$  of the matroid and observe that its Möbius transform only has 4 non-zero values:  $-1$  for circuits in  $\mathcal{C}$  and  $+1$  for  $C = N$ . In particular, by Corollary 2, (15) has the following simple form in the characteristic-imset mode:

$$c(abcd) + c(ace) + c(bde) - c(abcde) \leq 2.$$

Example 9 indicated a way one can search for connected matroids, and, thus, for facet-defining inequalities to be used in the ILP approach to BN structure learning. Graphic matroids are common examples of matroids; but there are many matroids which are not graphic, like the uniform matroid from Example 3.

To utilize fully the matroid-based approach some computer scientists may take the following exhaustive “brute-force” approach: given a (presumably) small cluster  $C$ ,  $|C| \geq 2$  generate by means of a computer all (permutation) types of classes  $\mathcal{C}$  of inclusion-incomparable subsets of  $C$  such that  $\forall a, b \in C, a \neq b$ , a set  $D \in \mathcal{C}$  exists with  $a, b \in D$ . Then one can check (again with the help of a computer) which of them satisfy the circuit elimination axiom. In this way one gets all types of connected matroids on  $C$  and can transform them into facet-defining inequalities for the family-variable polytope or for the characteristic imset polytope.

Other people may prefer to search in the literature on matroid theory. Indeed, researcher in this area have generated various catalogues of (types of) matroids on small ground sets; see, for example (Mayhew, Royle, 2008).

## 6 CONCLUSIONS

Theorem 3 implies that every connected matroid on a non-trivial cluster of BN variables induces a facet-defining inequality for the family-variable polytope; Corollary 2 says what is the form of that inequality in the context of the characteristic-imset polytope.

This is a quite general theoretical result because the well-known  $k$ -cluster inequalities, which play the key role in contemporary ILP approaches to BN structure learning, can be derived in this way. Specifically, they correspond to the prominent (connected) uniform matroids.

The significance of the paper is mainly theoretical: the area of statistical learning is related to a seemingly remote field in discrete mathematics, namely to matroid theory. Although matroids were previously known to have many applications in *combinatorial optimization*, this particular intimate link to BN structure learning could be surprising. The advantage of the matroid-based approach to learning is that the inequalities are easy to find and the verification that they are facet-defining is immediate since testing whether a matroid is connected is easy. The result is applicable in both ILP approaches to BN structure learning, that is, both in the context of the family-variable polytope and in the context of the characteristic-imset polytope.

However, the result also has some potential for practical future use because it may lead to bettering certain currently used algorithms. Let me recall in more detail the original motivation, which was the ILP approach to BN structure learning. I have in mind the *cutting plane method* where one solves an ILP optimization problem by the method of iterative reduction of the feasible set. The solution to a *linear relaxation* problem, which is a (non-integer) linear program with a larger feasible set, specified by a small number of inequalities, is typically a fractional vector. The next step is to solve the *separation problem*, that is, to find an inequality from a reservoir of available inequalities (for example from the class of cluster inequalities) which cuts the current fractional solution from the true feasible region, which is the polytope defined as the convex hull of integer vectors in the feasible set, see (Wolsey, 1998, § 8.5)

From the point of view of computational efficiency, it is essential to find such inequality which approximates the polytope as close as possible near the current solution. This leads to the suggestion to look for the most violated inequalities by the current fractional solution; see also the heuristic justification in (Cussens, 2011, § 4.1).

The presented result broadens the reservoir of available facet-defining inequalities in the ILP approach to BN structure learning. In fact, it is claimed by Bartlett and Cussens in (2013, § 6) that the inequality (13) from Example 7 has appeared to be particularly useful in their experiments. Moreover, the other facet-defining inequalities for  $F$ , that is, those not based on matroids, have not appeared to be very useful. Their empirical observations are the basis for my hope that the matroid-based inequalities may bring some further progress in this area, perhaps even resulting in better future running times.

Nevertheless, let me emphasize that additional problems have to be solved to reach the practical applicability of

general matroid-based inequalities. More specifically, it is necessary to solve the corresponding separation problem, that is, to design a speedy algorithm for finding the (most) violated inequalities by a current (fractional) solution in the class of all general matroid-based inequalities. Thus, the next step towards the practical application of the matroid-based inequalities should be a proposal for such algorithm.

## Acknowledgements

The research on this topic has been supported by the grant GAČR n. 13-20012S. I am indebted to my colleague Fero Matúš for giving me some guidance in matroid theory.

## References

- M. Bartlett, J. Cussens (2013). Advances in Bayesian network learning using integer programming. In *Uncertainty in Artificial Intelligence 29*, AUAI Press, 182-191.
- A. Barvinok (2002). *A Course in Convexity*. Graduate Studies in Mathematics 54, Providence: American Mathematical Society.
- R.R. Bouckaert (1995). Bayesian belief networks: from construction to evidence. PhD thesis, University of Utrecht.
- D.M. Chickering (2002). Optimal structure identification with greedy search. *Journal of Machine Learning Research* 3:507-554.
- J. Cussens (2010). Maximum likelihood pedigree reconstruction using integer programming. In *Proceedings of the Workshop on Constraint Based Methods for Bioinformatics (WCBMB)*, 9-19.
- J. Cussens (2011). Bayesian network learning with cutting planes. In F. Cozman, A. Pfeffer (eds.) *Uncertainty in Artificial Intelligence 27*, AUAI Press, 153-160.
- J. Cussens, M. Bartlett (2015). GOBNILP software. Available at [www.cs.york.ac.uk/aig/sw/gobnilp](http://www.cs.york.ac.uk/aig/sw/gobnilp).
- J. Cussens, D. Haws, M. Studený (2015). Polyhedral aspects of score equivalence in Bayesian network structure learning. Submitted to *Mathematical Programming A*, also available at [arxiv.org/abs/1503.00829](http://arxiv.org/abs/1503.00829).
- C.P. de Campos, Q. Ji (2011). Efficient structure learning Bayesian networks using constraints. *Journal of Machine Learning Research* 12:663-689.
- D. Heckerman, D. Geiger, D.M. Chickering (1995). Learning Bayesian networks: the combination of knowledge and statistical data. *Machine Learning* 20:194-243.
- R. Hemmecke, S. Lindner, M. Studený (2012). Characteristic imsets for learning Bayesian network structure. *International Journal of Approximate Reasoning* 53:1336-1349.
- T. Jaakkola, D. Sontag, A. Globerson, M. Meila (2010).

Learning Bayesian network structure using LP relaxations. In Y.W. Teh, M. Titterton (eds.) JMLR Workshop and Conference Proceedings 9: AISTATS 2010, 358-365.

S.L. Lauritzen (1996). *Graphical Models*. Oxford: Clarendon Press.

D. Mayhew, G.F. Royle (2008). Matroids with nine elements. *Journal of Combinatorial Theory B* **98**:415-431.

R.E. Neapolitan (2004). *Learning Bayesian Networks*. Upper Saddle River: Pearson Prentice Hall.

H.Q. Nguyen (1978). Semimodular functions and combinatorial geometries. *Transaction of the American Mathematical Society* **238**:355-383.

J.G. Oxley (1992). *Matroid Theory*. Oxford: Oxford University Press.

G.E. Schwarz (1978). Estimation of the dimension of a model. *Annals of Statistics* **6**:461-464.

T. Silander, P. Myllymäki (2006). A simple approach for finding the globally optimal Bayesian network structure. In R. Dechter, T. Richardson (eds.) *Uncertainty in Artificial Intelligence 22*, AUAI Press, 445-452.

M. Studený (2005). *Probabilistic Conditional Independence Structures*. London: Springer.

M. Studený, D.C. Haws (2013). On polyhedral approximations of polytopes for learning Bayesian networks. *Journal of Algebraic Statistics* **4**:59-92.

M. Studený, D. Haws (2014). Learning Bayesian network structure: towards the essential graph by integer linear programming tools. *International Journal of Approximate Reasoning* **55**:1043-1071.

T. Verma, J. Pearl (1991). Equivalence and synthesis of causal models. In *Uncertainty in Artificial Intelligence 6*, Elsevier, 220-227.

L.A. Wolsey (1998). *Integer Programming*. New York: John Wiley.

G.M. Ziegler (1995). *Lectures on Polytopes*. New York: Springer.

---

# The Long-Run Behavior of Continuous Time Bayesian Networks

---

Liessman Sturlaugson and John W. Sheppard

Department of Computer Science

Montana State University

Bozeman, MT 59717

listurlaugson@gmail.com, john.sheppard@cs.montana.edu

## Abstract

The continuous time Bayesian network (CTBN) is a temporal model consisting of interdependent continuous time Markov chains (Markov processes). One common analysis performed on Markov processes is determining their long-run behavior, such as their stationary distributions. While the CTBN can be transformed into a single Markov process of all nodes' state combinations, the size is exponential in the number of nodes, making traditional long-run analysis intractable. To address this, we show how to perform "long-run" node marginalization that removes a node's conditional dependence while preserving its long-run behavior. This allows long-run analysis of CTBNs to be performed in a top-down process without dealing with the entire network all at once.

## 1 INTRODUCTION

Many problems in artificial intelligence require reasoning about complex systems. One important and challenging type of system is one that changes in time. Temporal modeling and reasoning present additional challenges in representing the system's dynamics while efficiently and accurately inferring the system's behavior through time. Continuous time Bayesian networks (CTBNs) were introduced by Nodelman et al. (2002) as a temporal model capable of representing and reasoning about finite- and discrete-state systems without uniformly discretizing time, as found with dynamic Bayesian networks (Murphy, 2002). CTBNs have since been applied in a wide variety of temporal domains, from medical prognosis (Gatti et al., 2011; Gatti, 2011) to network security (Xu & Shelton, 2008, 2010) and reliability modeling (Herbrich et al., 2007; Cao, 2011; Sturlaugson & Sheppard, 2015).

While a variety of algorithms exist for querying probabilities of nodes in a CTBN at a specific time given temporal

evidence, another useful type of query is to analyze a network's long-run behavior, i.e., the stationary distributions of a CTBN's nodes. None of the previous CTBN inference algorithms were specifically designed to solve this problem. This paper presents the first inference algorithms for efficiently computing the stationary distribution of nodes in a CTBN.

The paper is organized as follows. Section 2 provides the background for the rest of the paper. Section 3 gives the theory and algorithms for computing stationary distributions in CTBNs. In Section 4, we demonstrate the algorithms on three CTBNs. Section 5 contains the conclusion and future work.

## 2 BACKGROUND

In this section, we begin by describing Markov processes and their long-run behavior. We then introduce the CTBN and discuss how combinations of nodes can be viewed as Markov processes.

### 2.1 MARKOV PROCESSES

Although its name draws on the parallels between the conditional independence encoded by Bayesian networks, the CTBN is functionally a factored Markov process. Therefore, we start with background on Markov processes.

#### 2.1.1 Definition

There are variations and extensions of the Markov process model, but the CTBN model uses the model described in this section. We refer to a finite-state, continuous-time Markov chain as a Markov process. In a Markov process, a system comprises a discrete set of states, and the system transitions probabilistically between these states. The difference between a Markov process and a Markov chain is that each transition occurs after a real-valued, exponentially distributed sojourn time, which is the time it remains in a state before transitioning. The parameters determining the sojourn times and the transition probabilities are en-

coded in what is called an “intensity matrix.” If the intensity matrix is constant throughout the lifetime of the system, we refer to the Markov process as “homogeneous.” Formally, we define a Markov process as follows.

**Definition 2.1** (Markov Process). *A finite-state, continuous-time, homogeneous Markov process  $X$  with a state space of size  $n$  is defined by an initial probability distribution  $P_X^0$  over the  $n$  states and an  $n \times n$  transition intensity matrix*

$$\mathbf{Q}_X = \begin{pmatrix} -q_{1,1}^X & q_{1,2}^X & \cdots & q_{1,n}^X \\ q_{2,1}^X & -q_{2,2}^X & \cdots & q_{2,n}^X \\ \vdots & \vdots & \ddots & \vdots \\ q_{n,1}^X & q_{n,2}^X & \cdots & -q_{n,n}^X \end{pmatrix}$$

in which each entry  $q_{i,j}^X \geq 0$  for  $i \neq j$  gives the transition rate of the process moving from state  $i$  to state  $j$ , and each entry  $q_{i,i}^X = \sum_j q_{i,j}^X$  is the parameter for an exponential distribution, determining the sojourn times for the process to remain in state  $i$ . For notational shorthand, the size of the state-space of  $X$  will be denoted as  $|X|$ .

The value  $q_{i,i}^X$  gives the rate at which the system leaves state  $x_i$ , while the value  $q_{i,j}^X$  gives the rate at which the system transitions from state  $x_i$  to state  $x_j$ . Let  $X(t)$  denote the state of  $X$  at time  $t$ . For  $i \neq j$ , we have that

$$\lim_{h \rightarrow 0^+} \frac{P(X(t+h) = x_j | X(t) = x_i)}{h} = q_{i,j}^X,$$

while  $q_{i,i}^X = \sum_{j \neq i} q_{i,j}^X$ . With the diagonal entries constrained to be non-positive, the probability density function for the process remaining in state  $i$  is given by  $q_{i,i}^X \exp(-q_{i,i}^X t)$ , with  $t$  being the amount of time spent in state  $i$ , making the probability of remaining in a state decrease exponentially with respect to time. The expected sojourn time for state  $i$  is  $1/|q_{i,i}^X|$ . The transition probabilities from state  $i$  to state  $j$  can be calculated as  $\theta_{i,j}^X = q_{i,j}^X/q_{i,i}^X$ . Because the sojourn time uses the exponential distribution, which is “memory-less,” the Markov process model exhibits the Markov property, namely, that all future states of the process are independent of all past states of the process given its present state. In other words, for  $0 < s < t < \infty$ ,

$$P(X(t+h)|X(t), X(s)) = P(X(t+h)|X(t)).$$

Rather than looking at the Markov process as a whole, we can also consider subsets of states, which we refer to as a subsystem. Formally, a subsystem  $S$  defines the behavior of a subset of states of a full Markov process  $X$ . The intensity matrix  $\mathbf{Q}_S$  of the subsystem  $S$  is formed from the entries of  $\mathbf{Q}_X$  that correspond to the states in  $S$ .

### 2.1.2 Stationary Distributions of Markov Processes

A common analysis of Markov processes is to consider their long-run behavior. In particular, we can consider the

stationary distribution  $\pi_X = \{\pi_1^X, \dots, \pi_n^X\}$  of the process, where

$$\pi_i^X = \lim_{t \rightarrow \infty} P(X(t) = i).$$

Assuming  $\mathbf{Q}_X$  is non-singular, we can compute the stationary distribution by setting up the system of equations

$$\pi_X = \mathbf{Q}_X \pi_X \quad (1)$$

with the added constraint  $\sum_{i=1}^n \pi_i^X = 1$  (Taylor & Karlin, 1998). The complexity of solving for  $\pi_X$  is determined by  $n$ , the number of states in  $X$ .

The state convergence properties of a Markov process can be analyzed from the stationary distribution. As an application used in the paper, this could be the expected long-term availability of a system. Changes to the model could be tested to optimize the reliability of the system (e.g, what is the minimum reliability of each component to guarantee the target reliability of the entire system). Stationary distributions of Markov processes have been used to analyze long-term trends in meteorology, economics, sociology, biology, immunology—just to name a few. As a factored Markov process, the CTBN can be used wherever a Markov process is applicable, while allowing the model to become more powerful and flexible through its factored representation.

## 2.2 CONTINUOUS TIME BAYESIAN NETWORKS

The CTBN was first introduced in Nodelman et al. (2002) and then further developed in Nodelman (2007) as a continuous-time probabilistic graphical model.

### 2.2.1 Definition

The motivation behind CTBNs is to factor a Markov process in much the same way that a Bayesian network factors a joint probability distribution. Instead of conditional probabilities, the CTBN uses conditional Markov processes. The CTBN is defined formally as follows.

**Definition 2.2** (Continuous Time Bayesian Network). *Let  $\mathbf{X}$  be a set of Markov processes  $\{X_1, X_2, \dots, X_n\}$ , where each process  $X_i$  has a finite number of discrete states. Formally, a continuous time Bayesian network  $\mathcal{N} = \langle \mathcal{B}, \mathcal{G} \rangle$  over  $\mathbf{X}$  consists of two components. The first is an initial distribution denoted  $P_{\mathbf{X}}^0$  over  $\mathbf{X}$  specified as a Bayesian network  $\mathcal{B}$ . This distribution  $P_{\mathbf{X}}^0$  is only used for determining the initial state of the process. The second is a continuous-time transition model  $\mathcal{G}$ , which describes the evolution of the process from its initial distribution.  $\mathcal{G}$  is represented as a directed graph with nodes  $X_1, X_2, \dots, X_n$ . Let  $\text{Pa}(X)$  denote the set of parents of  $X$  in  $\mathcal{G}$ , and let  $\text{Ch}(X)$  denote the set of children of  $X$  in  $\mathcal{G}$ . Let  $\mathbf{pa}_X$  denote the set of all combinations of state instantiations to  $\text{Pa}(X)$ , and let  $\langle pa_X \rangle \in \mathbf{pa}_X$ . A set of*

conditional intensity matrices (CIMs), denoted  $\mathbf{Q}_{X|\mathbf{Pa}(X)}$ , is associated with each  $X \in \mathbf{X}$  and comprises matrices  $\mathbf{Q}_{X|\langle pa_X \rangle} \forall \langle pa_X \rangle \in \mathbf{pa}_X$

For example, suppose we have a two-node CTBN with dependencies as  $A \rightleftharpoons B$ . Nodes  $A$  and  $B$  each have two states, with conditional intensity matrices as follows.

$$\mathbf{Q}_{A|b_0} = \begin{matrix} & a_0 & a_1 \\ a_0 & \begin{pmatrix} -1 & 1 \\ 2 & -2 \end{pmatrix} \end{matrix}, \mathbf{Q}_{A|b_1} = \begin{matrix} & a_0 & a_1 \\ a_1 & \begin{pmatrix} -3 & 3 \\ 4 & -4 \end{pmatrix} \end{matrix}$$

$$\mathbf{Q}_{B|a_0} = \begin{matrix} & b_0 & b_1 \\ b_0 & \begin{pmatrix} -5 & 5 \\ 6 & -6 \end{pmatrix} \end{matrix}, \mathbf{Q}_{B|a_1} = \begin{matrix} & b_0 & b_1 \\ b_1 & \begin{pmatrix} -7 & 7 \\ 8 & -8 \end{pmatrix} \end{matrix}$$

Nodes  $A$  and  $B$  are two distinct but interdependent subsystems of a larger Markov process.

### 2.2.2 Amalgamation

While we have shown that the CTBN is able to represent a Markov process as a set of interdependent subsystems, it is also useful to show how the subsystems of a CTBN can be merged together into ‘‘supernodes’’ containing the dynamics of multiple subsystems (Nodelman et al., 2002).

First we introduce additional notation for specific state instantiations and sets of state instantiations. Let  $\langle pa_{X \setminus Y} \rangle$  denote the state instantiation  $\langle pa_X \rangle$  excluding any state of  $Y$  (this changes  $\langle pa_X \rangle$  only if  $Y$  is a parent of  $X$ ). Then  $\mathbf{Q}_{X|\langle pa_{X \setminus Y} \rangle, Y}$  is the set of conditional intensity matrices that are dependent on the state instantiation  $\langle pa_{X \setminus Y} \rangle$  and each state of  $Y$ ,

$$\mathbf{Q}_{X|\langle pa_{X \setminus Y} \rangle, Y} = \{\mathbf{Q}_{X|\langle pa_{X \setminus Y} \rangle, y} | y \in Y\}.$$

Let  $\mathbf{pa}_{X \setminus Y}$  denote the set of all combinations of state instantiations to  $\mathbf{Pa}(X)$  excluding any state of  $Y$  (again, this changes  $\mathbf{pa}_X$  only if  $Y$  is a parent of  $X$ ).

The process involves combining sets of conditional intensity matrices from two different nodes,  $\mathbf{Q}_{X|\langle pa_{X \setminus Y} \rangle, Y}$  and  $\mathbf{Q}_{Y|\langle pa_{Y \setminus X} \rangle, X}$ , and forming a new conditional intensity matrix  $\mathbf{Q}_{XY|\langle pa_{XY} \rangle}$ , where  $\langle pa_{XY} \rangle = \langle pa_{X \setminus Y} \rangle \cup \langle pa_{Y \setminus X} \rangle$ . That is, the state instantiations for the parents of  $X$  and  $Y$  are combined, excluding the state instantiations for  $X$  and  $Y$ . The state instantiations for  $X$  and  $Y$  are excluded from  $\langle pa_{XY} \rangle$  because  $\mathbf{Q}_{XY|\langle pa_{XY} \rangle}$  will be defined over all state combinations of  $X$  and  $Y$ .

Let  $q_{i,j}^X$  be entry  $i, j$  of  $\mathbf{Q}_{X|\langle pa_X \rangle, y_k}$ , and let  $q_{k,l}^Y$  be entry  $k, l$  of  $\mathbf{Q}_{Y|\langle pa_Y \rangle, x_i}$ . The combined CIM  $\mathbf{Q}_{XY|\langle pa_{XY} \rangle}$  is the matrix defined over the states  $(x_i, y_k)$ , with the entries populated as follows.

---

#### Algorithm 1 Amalgamate two nodes of a CTBN.

---

Amalgamate( $X, Y$ )

```

1:  $\mathbf{Q}_{XY|\mathbf{Pa}(XY)} \leftarrow \emptyset$ 
2: for  $\langle pa_{X \setminus Y} \rangle \in \mathbf{pa}_{X \setminus Y}; \langle pa_{Y \setminus X} \rangle \in \mathbf{pa}_{Y \setminus X}$ 
3:    $\mathbf{Q}_{XY} \leftarrow \mathbf{0}$ 
4:   for  $i, j = 1, \dots, |X|; l, k = 1, \dots, |Y|$ 
5:      $\mathbf{Q}_X \leftarrow \mathbf{Q}_{X|\langle pa_{X \setminus Y} \rangle, x_i}$ 
6:      $\mathbf{Q}_Y \leftarrow \mathbf{Q}_{Y|\langle pa_{Y \setminus X} \rangle, y_k}$ 
7:     if  $i = j \wedge k = l$ 
8:        $q_{(i,j),(k,l)}^{XY} \leftarrow q_{i,j}^X + q_{k,l}^Y$ 
9:     else if  $i = j \wedge k \neq l$ 
10:       $q_{(i,j),(k,l)}^{XY} \leftarrow q_{k,l}^Y$ 
11:     else if  $i \neq j \wedge k = l$ 
12:       $q_{(i,j),(k,l)}^{XY} \leftarrow q_{i,j}^X$ 
13:     end if
14:   end for
15:    $\mathbf{Q}_{XY|\langle pa_{XY} \rangle} \leftarrow \mathbf{Q}_{XY}$ 
16:    $\mathbf{Q}_{XY|\mathbf{Pa}(XY)} \leftarrow \mathbf{Q}_{XY|\mathbf{Pa}(XY)} \cup \{\mathbf{Q}_{XY|\langle pa_{XY} \rangle}\}$ 
17: end for
18: return  $\mathbf{Q}_{XY|\mathbf{Pa}(XY)}$ 

```

---

$$q_{(i,j),(k,l)}^{XY} = \begin{cases} q_{i,j}^X & \text{if } i \neq j \text{ and } k = l \\ q_{k,l}^Y & \text{if } i = j \text{ and } k \neq l \\ q_{i,j}^X + q_{k,l}^Y & \text{if } i = j \text{ and } k = l \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

The CIM  $\mathbf{Q}_{XY|\langle pa_{XY} \rangle}$  defines the simultaneous dynamics of  $X$  and  $Y$ , given that their parents are in states  $\langle pa_{XY} \rangle$ . Thus, the state-space of  $XY$  is the Cartesian product of the states of  $X$  and  $Y$ , making  $\mathbf{Q}_{XY|\langle pa_{XY} \rangle}$  an  $|X||Y| \times |X||Y|$  matrix.

**Definition 2.3** (Amalgamation). *Amalgamation takes two nodes  $X$  and  $Y$  and replaces them with node  $XY$ , having the set of conditional intensity matrices  $\mathbf{Q}_{XY|\mathbf{Pa}(XY)}$  as formed by combining  $\mathbf{Q}_{X|\langle pa_{X \setminus Y} \rangle, Y}$  and  $\mathbf{Q}_{Y|\langle pa_{Y \setminus X} \rangle, X} \forall \langle pa_{X \setminus Y} \rangle \in \mathbf{pa}_{X \setminus Y}$  and  $\forall \langle pa_{Y \setminus X} \rangle \in \mathbf{pa}_{Y \setminus X}$  according to Equation 2. Amalgamation can be viewed as a multiplication operation over sets of conditional intensity matrices and is denoted  $\mathbf{Q}_{XY|\mathbf{Pa}(XY)} = \mathbf{Q}_{X|\mathbf{Pa}(X)} \times \mathbf{Q}_{Y|\mathbf{Pa}(Y)}$ .*

Amalgamation takes two nodes and combines all of their CIMs, producing a set of CIMs that are conditioned on the combined parent states of  $X$  and  $Y$ . Thus, the set  $\mathbf{Q}_{XY|\mathbf{Pa}(XY)}$  contains  $\prod_{Z \in \mathbf{Pa}(XY)} |Z|$  conditional intensity matrices.

Algorithm 1 shows the pseudocode for amalgamating two nodes of a CTBN. Line 1 initializes the empty set of conditional intensity matrices for the amalgamated node. Lines 2-18 iterate over all combinations of parent state instantiations of  $X$  and  $Y$ , excluding the state of  $X$  and  $Y$ . Line 3 initializes the conditional intensity matrix to be populated. Lines 4-14 iterate over the state combinations of  $X$

and  $Y$ . Lines 5-6 assign the conditional intensity matrices to temporary variables for simpler notation. Lines 7-13 populate the parameters of the conditional intensity matrix initialized in line 3 per Equation 2. Lines 15-17 add the conditional intensity matrix to the set of conditional intensity matrices of the amalgamated node, which is returned in Line 19.

**Definition 2.4** (Full Joint Intensity Matrix). *The full joint intensity matrix of a CTBN is the matrix resulting from amalgamating all nodes of the CTBN,*

$$\mathbf{Q} = \prod_{X \in \mathcal{N}} \mathbf{Q}_{X|\text{Pa}(X)}.$$

The size of  $\mathbf{Q}$  is  $n \times n$ , where  $n = \prod_{X \in \mathcal{N}} |X|$ .

For example, the full joint intensity matrix from the CTBN in Section 2.2.1 is as follows.

$$\mathbf{Q}_{AB} = \begin{matrix} & (a_0, b_0) & (a_0, b_1) & (a_1, b_0) & (a_1, b_1) \\ \begin{matrix} (a_0, b_0) \\ (a_0, b_1) \\ (a_1, b_0) \\ (a_1, b_1) \end{matrix} & \begin{pmatrix} -6 & 5 & 1 & 0 \\ 6 & -9 & 0 & 3 \\ 2 & 0 & -9 & 7 \\ 0 & 4 & 8 & -12 \end{pmatrix} \end{matrix}$$

### 3 NODE MARGINALIZATION IN THE LIMIT

Now we want to address the problem of computing stationary distributions for nodes in a CTBN  $\mathcal{N}$ . Formally, we want to compute  $\pi_X$  for  $X \in \mathbf{X}$ . We can calculate the stationary distribution for a node  $X$  having no parents in  $\mathcal{N}$  by simply using the approach of Equation 1 on the single intensity matrix of  $X$  (provided that  $X$  is irreducible). For nodes with dependencies (which is the point of the CTBN model), each node's stationary distribution depends on all ancestors in the network. In the worst case, a node could have all other nodes as ancestors (one of our experiments is a case of this). But as we have shown, the number of equations is exponential in the size of the network when working with the full joint intensity matrix directly.

We need to perform node marginalization so as to remove a node's dependence on its parents. This will allow us to contain the problem to individual subnetworks and not the entire network. Marginalization methods for CTBNs have been developed in the past, most notably expectation propagation (Nodelman et al., 2005) and belief propagation (El-Hay et al., 2010). Both of these methods approximate a node's unconditional intensity matrix; however, each of the unconditional intensity matrices by these methods are computed for a specific interval of constant evidence. They are not intended to describe the dynamics of a node as  $t \rightarrow \infty$ ,

which is what we need if we are to compute stationary distributions. The remainder of this section develops a novel CTBN node marginalization method that computes a long-run unconditional intensity matrix.

#### 3.1 THEORY

The key to computing the stationary distributions of nodes in the CTBN is to compute stationary distributions of subsystems of a Markov process. This allows us to work with subsets of nodes, instead of dealing with  $\mathbf{Q}$  all at once. Let  $S$  be the starting subsystem and  $D$  be the destination subsystem. We now want to compute the rate at which  $S$  transitions to  $D$  in the limit. Formally, we want a tractable way to evaluate

$$q_{S,D}^X = \lim_{t \rightarrow \infty} \lim_{h \rightarrow 0^+} \frac{P(X(t+h) \in D | X(t) \in S)}{h}. \quad (3)$$

**Theorem 3.1.** *For a Markov process with disjoint subsystems  $S$  and  $D$  and  $q_{S,D}^X$  as defined above, then*

$$q_{S,D}^X = \frac{1}{Z} \sum_{i \in S} \pi_i^X \sum_{j \in D} q_{ij}^X$$

where  $Z = \sum_{i \in S} \pi_i^X$ .

*Proof.* Because  $S$  is a set of multiple states when conditioning on  $X(t) \in S$ , we must weight each state  $i$  in  $S$  by the probability of being in state  $i$  at time  $t$  and renormalize.

$$\begin{aligned} \lim_{t \rightarrow \infty} \lim_{h \rightarrow 0^+} \frac{P(X(t+h) \in D | X(t) \in S)}{h} &= \\ \lim_{t \rightarrow \infty} \lim_{h \rightarrow 0^+} \frac{1}{\sum_{i \in S} P(X(t) = i)} \sum_{i \in S} P(X(t) = i) &\times \\ \sum_{j \in D} \frac{P(X(t+h) = j | X(t) = i)}{h} &= \\ \lim_{t \rightarrow \infty} \frac{1}{\sum_{i \in S} P(X(t) = i)} \sum_{i \in S} P(X(t) = i) &\times \\ \sum_{j \in D} \lim_{h \rightarrow 0^+} \frac{P(X(t+h) = j | X(t) = i)}{h} &= \\ \lim_{t \rightarrow \infty} \frac{1}{\sum_{i \in S} P(X(t) = i)} \sum_{i \in S} P(X(t) = i) \sum_{j \in D} q_{ij}^X &= \\ \frac{1}{\sum_{i \in S} \pi_i^X} \sum_{i \in S} \pi_i^X \sum_{j \in D} q_{ij}^X &= \\ \frac{1}{Z} \sum_{i \in S} \pi_i^X \sum_{j \in D} q_{ij}^X & \quad \square \end{aligned}$$

**Corollary 3.2.** *Suppose that a Markov process  $X$  comprises  $n$  disjoint subsystems  $\{S_1, \dots, S_n\}$ . Then  $q_{S_i, S_i}^X = \sum_{j=1, j \neq i}^n q_{S_i, S_j}^X$ .*

---

**Algorithm 2** Compute long-run unconditional intensity matrix of a node.

---

*MarginalizeNode*( $Fam_X$ )

```

1: for  $i = 1, \dots, |X|$ 
2:   for  $j = 1, \dots, |X|$  s.t.  $j \neq i$ 
3:      $Z \leftarrow \sum_{k \in S_i} \pi_k^X$ 
4:      $q_{i,j}^X \leftarrow \frac{1}{Z} \sum_{k \in S_i} \pi_k^X \sum_{l \in D_j} q_{k,l}^{Fam_X}$ 
5:   end for
6:    $q_{i,i}^X = \sum_{j=1, j \neq i}^{|X|} q_{i,j}^X$ 
7: end for
8: return  $Q_X$ 

```

---

*Proof.* This follows from the definition of a Markov process, which constrains  $q_{i,i}^X = \sum_{j \neq i} q_{i,j}^X$ . In other words, if we know the rate at which the process leaves one subsystem and enters every other subsystem, we also know the rate with which the process leaves the subsystem.  $\square$

### 3.2 ALGORITHMS

Now we apply this idea of computing long-run transition rates for Markov process subsystems to amalgamated nodes in a CTBN. In this case, we take the subsystems to be the states of a child node in an amalgamation of the child and its parents. After computing the long-run transition rates for the subsystems, we can construct an unconditional intensity matrix for the child node.

Algorithm 2 computes a long-run unconditional intensity matrix for node  $X$  from a set of amalgamated nodes  $Fam_X$  that includes  $X$  and all parents of  $X$ . The variables  $i$  and  $j$  iterate over the rows and columns, respectively, for the unconditional intensity matrix of  $X$ . Line 3 computes the normalization constant. The sets  $S_i$  and  $D_j$  are the states in  $Fam_X$  that include state  $i$  and state  $j$  of  $X$ , respectively. Line 4 computes the long-run transition rate of node  $X$  from state  $i$  to state  $j$ , according to Theorem 3.1. Line 6 computes the long-run sojourn rate of state  $i$  of node  $X$ , according to Corollary 3.2. Line 8 returns the long-run unconditional intensity matrix of  $X$  that are populated by entries  $q_{i,j}^X$ . The complexity of Algorithm 2 is dominated by the computation of the stationary distribution. Assuming  $Q_X$  is non-singular, its stationary distribution can be computed in  $O(n^3)$ .

Now that we can compute a long-run unconditional intensity matrix for a node, we can break the dependence of the child on its parents. The unconditional intensity matrix computed for the child will already incorporate the stationary distribution of the parents. We can repeat the process in a top-down fashion through the network, computing the stationary distributions of every node in the network without having to deal with the entire network all at once. Algorithm 3 calculates the long-run unconditional intensity matrices for all of the nodes in a CTBN.

---

**Algorithm 3** Compute long-run unconditional intensity matrices of a CTBN.

---

*MarginalizeCTBN*( $\mathcal{G}$ )

```

1:  $\mathcal{G}' \leftarrow \text{CollapseCycles}(\mathcal{G})$ 
2: repeat until termination
3:    $L_1 \leftarrow \bigcup_{X \in \mathcal{G}'} X$  s.t.  $\text{Pa}(X) = \text{null}$ 
4:    $L_2 \leftarrow \bigcup_{X \in L_1} \text{Ch}(X)$  s.t.  $\text{Pa}(\text{Ch}(X)) \subseteq L_1$ 
5:   if  $L_2 = \text{null}$  then terminate
6:   for  $X \in L_2$ 
7:      $Fam_X \leftarrow X$ 
8:     for  $Y \in \text{Pa}(X)$ 
9:        $Fam_X \leftarrow \text{Amalgamate}(Fam_X, Y)$ 
10:    end for
11:     $Q_{X'} \leftarrow \text{MarginalizeNode}(Fam_X)$ 
12:    for  $Y \in \text{Pa}(X)$ 
13:      remove edge  $(Y, X)$  from  $\mathcal{G}'$ 
14:    end for
15:     $Q_X \leftarrow Q_{X'}$ 
16:  end for
17: end repeat

```

---

We need to turn  $\mathcal{G}$  into a directed acyclic graph (DAG) from which we can divide the graph into top-to-bottom levels. The behavior of a node depends on all of its ancestors, thus to create a DAG we need to amalgamate all the nodes of each cycle. In the next section we will show an approximation step that avoids collapsing cycles when the cycles themselves are too large for their amalgamation to be tractable.

In Algorithm 3, line 1 collapses the cycles in the  $\mathcal{G}$  by amalgamating all of the nodes in each cycle. Lines 2-17 iterate over the levels of the DAG. Lines 3-4 find all of the 2nd-level nodes, i.e., nodes with no other ancestors than their immediate parents. If there are no more 2nd-level nodes, then all of the nodes have been marginalized (no nodes have parents), and line 5 terminates the loop. Lines 6-16 iterate over all of the 2nd-level nodes. Lines 7-10 amalgamate each 2nd-level node with all of its parents. Line 11 computes the long-run unconditional intensity matrix for each 2nd-level node. Lines 12-14 remove the dependency of the node on its parents, and line 15 updates the node's set of intensity matrices with the single intensity matrix from line 11. At the conclusion of the algorithm, the nodes of  $\mathcal{N}$  are individual unconditional Markov processes. Note that the stationary distributions are computed along the way by Algorithm 2. The complexity of the algorithm is dominated by the maximum number of parents of any node (analogous to tree-width in Bayesian network inference).



### 3.3 APPROXIMATION FOR CYCLES

One difficulty of node marginalization is that the dynamics of a node depend on all of its ancestors. If the network is a directed acyclic graph (DAG), then we can marginalize each level in succession, and the complexity of isolation depends on the number of immediate parents to each node. However, cycles are allowed in CTBNs. When a cycle is introduced, every node in the cycle must be included to marginalize any node in the cycle, because every node in the cycle is an ancestor of every other node in the cycle.

We can address this complication by adding an iterative step to our long-run node marginalization algorithm that avoids dealing with the entire cycle all at once. First, we identify the cycles and all of the nodes they comprise. Let  $\mathbf{X}_C$  denote the set of arbitrarily chosen nodes that cover all of the cycles (it is possible that a single node could cover multiple cycles). The set  $\mathbf{X}_C$  covers a cycle when at least one node in  $\mathbf{X}_C$  is part of the cycle.

For each  $X \in \mathbf{X}_C$ , we temporarily remove all incoming arcs. Previously, the node had a set of conditional intensity matrices, whereas now we need to replace it with one unconditional intensity matrix. While this unconditional intensity matrix depends on the dynamics of the parents that were just removed, we simply use an unconditional intensity matrix that is the average of the node’s conditional intensity matrices. Formally, for each  $X \in \mathbf{X}_C$ , we remove the incoming arcs to  $X$  and estimate an initial unconditional intensity matrix for  $X$  as

$$\hat{\mathbf{Q}}_X \leftarrow \frac{1}{|\mathbf{Q}_{X|\mathbf{Pa}(X)}|} \sum_{\mathbf{Q}_{X|\langle pa_X \rangle} \in \mathbf{Q}_{X|\mathbf{Pa}(X)}} \mathbf{Q}_{X|\langle pa_X \rangle}.$$

Once we have done this for every cycle, the graph becomes a DAG, and we run the *MarginalizeCTBN* algorithm as before.

Depending on the actual parameters, the resulting unconditional intensity matrices could be a poor approximation, because of how we estimated the unconditional intensity matrix of the nodes in  $\mathbf{X}_C$ . Now we can improve on our estimates for  $\hat{\mathbf{Q}}_X$  because, after the first iteration, we have an unconditional intensity matrix for every immediate parent of  $X$ . So we add back all of the incoming nodes of each  $X \in \mathbf{X}_C$  and call *MarginalizeNode* on each of these nodes. This results in updated estimates for each  $\hat{\mathbf{Q}}_X$  which now take into account an estimate of the dynamics of the parents of each  $X$ . Now we have the original DAG with updated unconditional intensity matrices for each  $X \in \mathbf{X}_C$ . We can call *MarginalizeCTBN* again.

This process continues to loop around the cycles until convergence. This process is analogous to loopy belief propagation in cyclic graphs, such as in Markov random fields

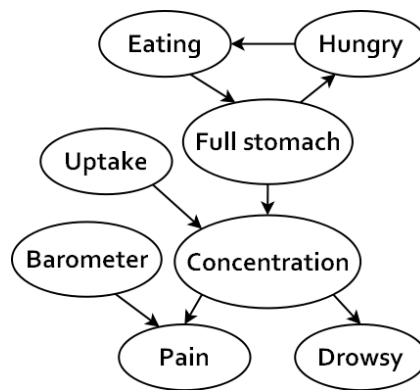


Figure 1: Drug effect network.

and in Bayesian networks in which the acyclic constraint has been relaxed (Koller & Friedman, 2009). The long-run unconditional intensity matrices are approximations in this case, because we have never viewed the cycle as a whole. On the other hand, if the cycles have too many nodes, we have kept the problem tractable.

## 4 EXPERIMENTS

We demonstrate the long-run marginalization methods on three networks—two synthetic networks and one real-world network. The two synthetic networks are small enough that we can compute the stationary distributions from the full joint intensity matrix. For the real-world network, we can approximate the stationary distributions by forward sampling the CTBN long enough into the future such that the samples will have converged to the stationary distribution.

### 4.1 DRUG EFFECT NETWORK

First, we used the drug effect network from Nodelman et al. (2002) as shown in Figure 1. The network is a toy model that shows the interaction of several variables on a patient’s pain and drowsiness.

We collapse the *Hungry*  $\rightarrow$  *Eating*  $\rightarrow$  *Full Stomach* cycle and marginalize *Concentration*. The unconditional intensity matrix of *Concentration* is

$$\mathbf{Q}_{Concentration} \approx \begin{pmatrix} -0.02 & 0.01 & 0.01 \\ 0.25 & -0.26 & 0.01 \\ 0.01 & 0.50 & -0.51 \end{pmatrix}.$$

We then marginalize *Pain* and *Drowsy*, which yields,

$$\mathbf{Q}_{Pain} \approx \begin{pmatrix} -0.56 & 0.56 \\ 0.28 & -0.28 \end{pmatrix}$$

and

$$\mathbf{Q}_{Drowsy} \approx \begin{pmatrix} -0.18 & 0.18 \\ 0.46 & -0.46 \end{pmatrix}.$$

The calculated stationary distributions from both the marginalized nodes and the full joint intensity matrix are

$$\begin{aligned}\pi_{\text{pain}} &\approx 0.336 \\ \pi_{\text{pain-free}} &\approx 0.664 \\ \pi_{\text{drowsy}} &\approx 0.713 \\ \pi_{\text{non-drowsy}} &\approx 0.287.\end{aligned}$$

However, using the full joint intensity matrix (brute-force method) required solving a system of 864 equations. Through our long-run node marginalization method, we needed to solve systems of 72, 12, and 4 equations. In other words, the complexity has been reduced by approximately a factor of 10.

## 4.2 RING NETWORK

This second experiment tests our long-run marginalization method on cyclic networks of varying length. For a ring network of size  $n$ , we construct the network by adding  $n$  three-state ( $s_0, s_1, s_2$ ) nodes and connecting them as follows:

$$X_1 \rightarrow X_2 \rightarrow \dots \rightarrow X_n \rightarrow X_1.$$

Let each  $q_{i,j}^k$  be an independent sample from a uniform distribution over the interval  $(0, 1)$ . The conditional intensity matrices for the nodes are defined as follows (for ease of definition,  $X_0$  and  $X_n$  denote the same node):

$$\begin{aligned}\mathbf{Q}_{X_k|X_{k-1}=s_0} &= \begin{pmatrix} -q_{1,1}^k & q_{1,1}^k & 0 \\ 0 & -q_{1,2}^k & q_{1,2}^k \\ q_{1,3}^k & 0 & -q_{1,3}^k \end{pmatrix}, \\ \mathbf{Q}_{X_k|X_{k-1}=s_1} &= \begin{pmatrix} -q_{2,1}^k & \frac{q_{2,1}^k}{2} & \frac{q_{2,1}^k}{2} \\ \frac{q_{2,2}^k}{2} & -q_{2,2}^k & \frac{q_{2,2}^k}{2} \\ \frac{q_{2,3}^k}{2} & \frac{q_{2,3}^k}{2} & -q_{2,3}^k \end{pmatrix}, \\ \mathbf{Q}_{X_k|X_{k-1}=s_2} &= \begin{pmatrix} -q_{3,1}^k & 0 & q_{3,1}^k \\ q_{3,2}^k & -q_{3,2}^k & 0 \\ 0 & q_{3,3}^k & -q_{3,3}^k \end{pmatrix}.\end{aligned}$$

We vary the length of the cycle from 3 nodes to 8 nodes and, for each cycle length, apply the approximate node marginalization method described in Section 3.3 and compare the accuracy to the results from the brute-force method. (Note that in this case the brute-force and exact node marginalization methods are identical, because the whole network is a cycle.) Because each network is generated with random parameters, we run a total of 100 trials for each cycle length and average the results. We keep track of the average number of iterations for the stationary distribution estimates to converge, and we compute the average KL-divergence of the stationary distributions results using the full joint intensity matrix from the results using the iterative node marginalization method. These results are shown in Table 1.

Table 1: Results for the ring networks.

| Cycle Length | Avg. Iterations to Converge | Average KL-Divergence |
|--------------|-----------------------------|-----------------------|
| 3            | 12.1                        | 3.1E-4                |
| 4            | 10.1                        | 5.5E-5                |
| 5            | 8.2                         | 2.1E-5                |
| 6            | 7.6                         | 1.7E-5                |
| 7            | 6.8                         | 1.9E-5                |
| 8            | 6.0                         | 1.4E-5                |

At least for these randomly generated networks, the iterative node marginalization method maintained accurate estimates of the stationary distributions, and the number of iterations to converge tended to decrease as the cycle grew. Notice that the error decreases as the cycle length increases. For these networks, at least, the dynamics of a node are most influenced by the immediate parent. The second-most influential node is the parent’s parent, and so on. As the length of the cycle increases, the influence of the “arc that completes the cycle” (whichever arc one chooses this to be) exerts less influence on the dynamics of the cycle as a whole. Therefore, temporarily removing an arc has a decreasing impact as the cycle becomes larger.

For a 3-node cycle, amalgamating the whole cycle will most likely still be a tractable approach. For longer cycle lengths, on the other hand, the applicability of the iterative node marginalization method becomes more critical. In our setup, every node added to the cycle triples the size of the full joint intensity matrices and hence the system of equations to solve.

## 4.3 CARDIAC ASSIST SYSTEM

Third, we compared the inference methods on a larger, real-world network. We used the model for a cardiac assist system (CAS), presented by Cao (2011), which is broadly used in the literature and based on a real-world system (Boudali et al., 2007; Portinale et al., 2010). Cao (2011) shows how the CTBN is able to encode Dynamic Fault Trees (DFTs), which are reliability models that use Boolean logic to combine series of lower-level failure events while preserving failure sequence information (Dugan et al., 1992). The intensity matrices of the CTBN are used to represent the gates available in the DFT, including AND, OR, warm spare (WSP), sequence enforcing (SEQ), probabilistic dependency (PDEP), and priority AND (PAND). Our model for this experiment is the DFT for the CAS system represented as a CTBN. Of the various repair policies evaluated by Cao (2011), we use the repair rate of  $\mu = 0.1$  (10 hours) for all components.

Figure 2 shows the network, while Table 2 gives the node names. In this model, we are interested in the stationary

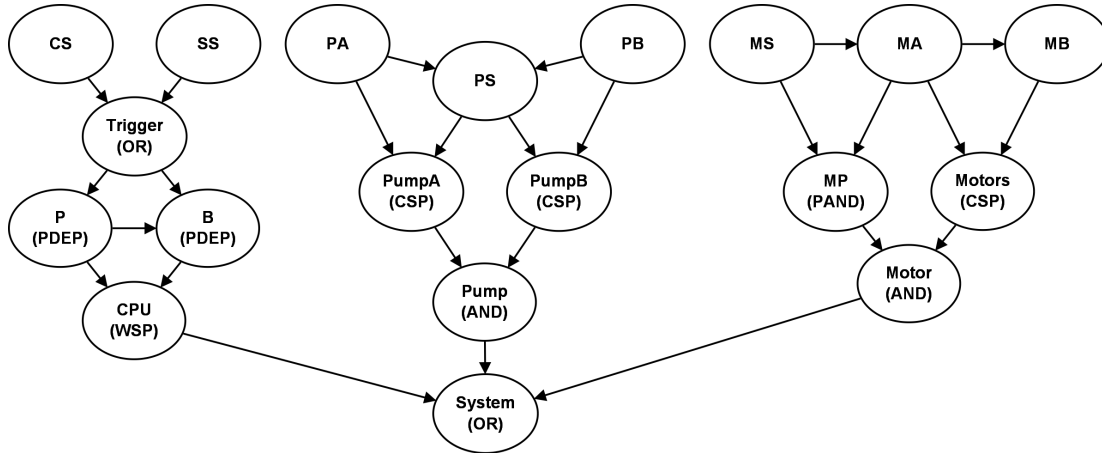


Figure 2: Cardiac assist system model.

distribution of the *System* node, i.e., in the long run, what proportion of the time will the *System* be operational? This corresponds to the operational availability of the subsystem.

Notice that the node in which we are most interested is a descendant of every other node in the network. Using our node marginalization method, we can compute the stationary distribution of the *System* node as

$$\begin{aligned}\pi_{\text{system-up}} &\approx 0.942 \\ \pi_{\text{system-down}} &\approx 0.058.\end{aligned}$$

If we had attempted to work with the full joint intensity matrix directly, we would be faced with solving a system of over 6.6 million equations. In other words, the brute-force method is intractable for this real-world network. Instead, our node marginalization method divided the network into a total of 14 subnetworks, with the two largest representing systems of only 20 equations.

Instead of trying to solve the system of equations for this large network, we can approximate the stationary distribution by forward sampling the CTBN and observing the convergence of the state probabilities. Because this is an approximation method, we ran multiple trials to quantify the average behavior of the approximation. We ran 100 trials and averaged the results. For each trial, we sampled the network so that we had 10K transitions for the *System* node. Because the dynamics of *System* depend on every other node, we had to sample transitions from all other nodes as well. By the time we had generated 10K transitions for *System*, we had generated more than 100K transitions on average for the other nodes. Our approximation of the stationary distribution of *System* from 10K transitions still resulted in a KL-divergence of  $7.2E-3$  on average. On the other hand, our node marginalization method computed the exact answer over 3 times faster on average.

Note that our node marginalization method produces an

Table 2: CAS component names.

| Abbreviation | Name                | Subsystem |
|--------------|---------------------|-----------|
| P            | primary CPU         | CPU       |
| B            | warm spare CPU      | CPU       |
| CS           | cross switch        | CPU       |
| SS           | system supervision  | CPU       |
| MA           | primary motor       | Motor     |
| MB           | cold spare motor    | Motor     |
| MS           | switching component | Motor     |
| PA           | pump A              | Pump      |
| PB           | pump B              | Pump      |
| PS           | cold shared pump    | Pump      |

Table 3: Expected sojourn times for CAS subsystems.

| Subsystem | MTBF (hrs) | MTTR (hrs) |
|-----------|------------|------------|
| CPU       | 154        | 9.38       |
| Pump      | 60K        | 5.00       |
| Motor     | 410M       | 6.50       |

other useful output. Because the method computes unconditional intensity matrices along the way, we can observe not only the stationary distributions of different nodes but their long-run expected sojourn times as well. For example, looking at the diagonal entries of the unconditional intensity matrices of *System*, we see that, in the long-run, the mean time between failures (MTBF) for the system is about 153 hours and the mean time to repair (MTTR) is about 9.37 hours. Which of the three subsystems contributes the most to these values? Because of the top-down marginalization process, we have already calculated the same values for each of three subsystems, summarized in Table 3.

We have identified that the *CPU* subsystem contributes the most to *System* failure, while the *Motor* subsystem, due to

its high reliability rates and its redundancy, very rarely contributes to *System* failures. The *Pump* subsystem is identified as the fastest to be repaired. From long-run analysis on both the stationary distributions and the expected sojourn times, we have efficiently identified and quantified the unreliability of the *CPU* subsystem for efforts to make the CAS more robust and reliable.

## 5 DISCUSSION

The experiments demonstrate the capability of the exact and approximate node marginalization methods developed in this paper. We started with two synthetic networks that were small enough to compute the stationary distributions using the traditional approach when viewing a CTBN as a Markov process via its full joint intensity matrix. With the drug effect network, we showed that the exact method computes the same values with a fraction of the computational complexity. With the cyclic network, the brute-force and exact methods become indistinguishable. We showed how an iterative variation of the exact node marginalization method can effectively approximate the stationary distributions of nodes in cycles without handling the entire cycle all at once. Lastly, we applied the node approximation method to a non-trivial real-world network. In this case, working with the full joint intensity matrix (the tradition, brute-force approach) is intractable. We compare our exact node marginalization method to an approximate method based on forward sampling. For this experiment, our node marginalization method is both more efficient and yields the exact answer instead of an approximation.

Our methods assume that the stationary distributions of individual subnetworks can be computed efficiently. Specifically, solving Equation 1 requires the matrix to be non-singular (i.e., that the Markov process be irreducible). For some CTBNs, this may not be the case, and some sub-systems of the process may not be irreducible. Our exact method breaks down in this case. However, note that the traditional approach also breaks down, because it is also based on Equation 1. As long as there exists a method to compute  $\pi$  for a subnetwork (or at least approximate  $\pi$ ), this vector can be used in our top-down and/or iterative node marginalization methods.

## 6 CONCLUSION

We have shown how to compute stationary distributions and long-run expected sojourn times for CTBNs tractably without working directly with the full joint intensity matrix. For CTBNs with long cycles, we have shown an iterative marginalization method that can be used to approximate the long-run behavior. To demonstrate the methods, we tested on three networks of varying complexity and showed the advantage of using our marginalization methods. Fu-

ture work involves analyzing the behavior of the iterative marginalization method, including research into network topologies and parameters that could make the approximation poor, as well as analyzing convergence properties such as proof of convergence.

## References

- Boudali, H., Crouzen, P., & Stoelinga, M. (2007). Dynamic fault tree analysis using input/output interactive Markov chains. In *37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks* (pp. 708–717).
- Cao, D. (2011). *Novel models and algorithms for systems reliability modeling and optimization*. Wayne State University.
- Dugan, J., Bavuso, S., & Boyd, M. (1992). Dynamic fault-tree models for fault-tolerant computer systems. *IEEE Transactions on Reliability*, 41(3), 363–377.
- El-Hay, T., Cohn, I., Friedman, N., & Kupferman, R. (2010). Continuous-time belief propagation. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*.
- Gatti, E. (2011). *Graphical models for continuous time inference and decision making*. Università degli Studi di Milano-Bicocca.
- Gatti, E., Luciani, D., & Stella, F. (2011). A continuous time Bayesian network model for cardiogenic heart failure. *Flexible Services and Manufacturing Journal*, 1–20.
- Herbrich, R., Graepel, T., & Murphy, B. (2007). Structure from failure. In *Proceedings of the 2nd USENIX workshop on tackling computer systems problems with machine learning techniques* (pp. 1–6).
- Koller, D., & Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques*. MIT Press.
- Murphy, K. (2002). *Dynamic Bayesian networks: representation, inference and learning*. University of California.
- Nodelman, U. (2007). *Continuous time Bayesian networks*. Stanford University.
- Nodelman, U., Koller, D., & Shelton, C. (2005). Expectation propagation for continuous time Bayesian networks. In *Proceedings of the Twenty-First Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-05)* (pp. 431–440). Arlington, Virginia: AUAI Press.
- Nodelman, U., Shelton, C., & Koller, D. (2002). Continuous time Bayesian networks. In *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence (UAI)* (pp. 378–387).

- Portinale, L., Raiteri, D., & Montani, S. (2010). Supporting reliability engineers in exploiting the power of dynamic Bayesian networks. *International Journal of Approximate Reasoning (IJAR)*, 51(2), 179–195.
- Sturlaugson, L., & Sheppard, J. W. (2015). Sensitivity analysis of continuous time Bayesian network reliability models. *SIAM/ASA Journal on Uncertainty Quantification*, 3(1), 346–369.
- Taylor, H., & Karlin, S. (1998). *An Introduction to Stochastic Modeling*. Academic Press.
- Xu, J., & Shelton, C. (2008). Continuous Time Bayesian Networks for Host Level Network Intrusion Detection. In W. Daelemans, B. Goethals, & K. Morik (Eds.), *Machine Learning and Knowledge Discovery in Databases* (Vol. 5212, pp. 613–627). Springer Berlin / Heidelberg.
- Xu, J., & Shelton, C. (2010, September). Intrusion detection using continuous time Bayesian networks. *Journal of Artificial Intelligence Research (JAIR)*, 39(1), 745–774.

---

# Online Bellman Residual Algorithms with Predictive Error Guarantees

---

**Wen Sun**  
Robotics Institute  
Carnegie Mellon University  
Pittsburgh, PA 15213  
wensun@cs.cmu.edu

**J. Andrew Bagnell**  
Robotics Institute  
Carnegie Mellon University  
Pittsburgh, PA 15213  
dbagnell@ri.cmu.edu

## Abstract

We establish a connection between optimizing the Bellman Residual and worst case long-term predictive error. In the online learning framework, learning takes place over a sequence of trials with the goal of predicting a future discounted sum of rewards. Our analysis shows that, together with a stability assumption, *any* no-regret online learning algorithm that minimizes Bellman error ensures small prediction error. No statistical assumptions are made on the sequence of observations, which could be non-Markovian or even adversarial. Moreover, the analysis is independent of the particular form of function approximation and the particular (stable) no-regret approach taken. Our approach thus establishes a broad new family of provably sound algorithms for Bellman Residual-based learning and provides a generalization of previous worst-case result for minimizing predictive error. We investigate the potential advantages of some of this family both theoretically and empirically on benchmark problems.

## 1 INTRODUCTION

*Reinforcement learning* (RL) is an online paradigm for optimal sequential decision making where an agent interacts with an environment, takes actions, receives rewards and tries to maximize its *long-term reward*, a discounted sum of all the rewards that will be received from now on. An important part of RL is policy evaluation, the problem of evaluating the expected long-term rewards of a fixed policy. *Temporal Difference* (TD) is a famous family of algorithms for policy evaluation. In practice, we are typically interested in complex problem domains (e.g., continuous state space RL) and function approximations (e.g., linear functions) are used for policy evaluation. However, it has been observed that when combined with function approxi-

mation, TD may diverge and lead to poor prediction. The *Residual Gradient* (RG) was proposed (Baird, 1995) to address these concerns. RG attempts to minimize the *Bellman Error* (BE) (see definition in Sec. 2), typically with linear function approximation, using stochastic gradient descent. Since then comparison between the family of TD algorithms and RG has received tremendous attention, although most of the analyses heavily rely on certain stochastic assumptions of the environment such as that the sequence of observations are Markovian or from a static Markov Decision Process (MDP). For instance Schoknecht and Merke (2003) showed that TD converges provably faster than RG if the value functions are presented by tabular form. Scherrer (2010) shows that Bellman Residual minimization enjoys a guaranteed performance while TD does not in general when states are sampled from arbitrary distributions that may not correspond to trajectories taken by the system. Experimentally, they also show that TD converges faster but may generate poor prediction when it is close to divergence.

Schapire and Warmuth (1996) and Li (2008) provided worst-case analysis of long-term predictive error for variants of the linear TD and RG under a non-probabilistic online learning setting. Their results rely on an elegant spectral analysis of a matrix that is related to **specific** update rules of the TD and RG algorithms under linear function approximation. Unfortunately, this approach makes it more difficult to extend their worst-case (assumption free) analysis to broader families of algorithms and representations that target the Bellman and Temporal Difference errors.

Following Schapire and Warmuth (1996) and Li (2008)'s online learning framework, we present a simple, general connection between long-term predictive error and no-regret online learning that attempts to minimize BE. The central idea is that methods such as RG should be fundamentally understood as online algorithms as opposed to standard gradient methods, and that one cannot simultaneously make consistent predictions in the sense of BE while doing a poor job in terms of long-run predictions. Similar to Schapire and Warmuth (1996) and Li (2008), our analysis does not rely on any statistical assumptions about the

underlying system. This allows us to analyze more difficult scenarios such as Markov Decision Process with transition probabilities changing over time or even with each transition chosen entirely adversarial. Our analysis generalizes to a broader class of functions to approximate the value function. Previous work from Robards et al. (2011) and Engel et al. (2005) explored the possibility of using non-linear function approximation, but to our knowledge no further analysis on the soundness with respect to prediction error are known.

Our analysis of the connection between online long-term reward prediction and no-regret online learning provides a unifying view of the relationship between prediction errors and BE and consequently suggests a broad new family of algorithms. Specifically, we present and analyze concrete examples of how to apply several well-known no-regret online algorithms such as *Online Gradient Descent* (OGD) from Zinkevich (2003), *Online Newton Step* (ONS) from Hazan et al. (2006) and *Online Frank Wolf* (OFW) from Hazan and Kale (2012) to online prediction of long-term rewards. Particularly, our analysis generalizes the RG algorithm from Baird (1995) in the following three aspects: (1) RG is a specific example of our family of algorithms that runs OGD on a sequence of BE loss functions, (2) RG can be naturally combined with more general function approximation such as functions in *Reproducing Kernel Hilbert Space* (RKHS), and (3) applying our analysis to RG provides asymptotically tighter bounds on the average prediction error of long-term rewards than that provided in Li (2008). We also find that ONS, which has no-regret rate of  $O(\log T/T)$ , has a faster convergence of the average prediction error of long-term rewards. With OFW, we are able to achieve sparse predictors under some conditions. We analyze these algorithms in detail in Sec. 4.

We emphasize that stability of online algorithms is essential for our results—the no-regret property can be shown by example to be insufficient to achieve low predictive error. We hence introduce the definition of *Online Stability* condition in Sec. 2, which intuitively measures the difference between two successive predictors. Our online stability condition is general enough such that most popular no-regret online algorithms naturally satisfy this condition and hence this condition does not severely limit the scope of no-regret online algorithms. Our analysis shows that the combination of the no-regret property and online stability is sufficient to promise small predictive error on the long-term rewards.

## 2 PRELIMINARIES

### 2.1 PROBLEM SETTING

We consider the sequential online learning model presented in Schapire and Warmuth (1996); Li (2008) where no sta-

tistical assumptions about the sequence of observations are made. The sequence of the observations forms a connected stream of states which can either be Markovian as typically assumed in RL problem settings or even adversarial. We define the observation at time step  $t$  as  $\mathbf{x}_t \in \mathbb{R}^n$ , which usually represents the features of the environment at  $t$ . Throughout the paper, we assume that feature vector  $\mathbf{x}$  is bounded as  $\|\mathbf{x}\|_2 \leq X, X \in \mathbb{R}^+$ . The corresponding reward at step  $t$  is defined as  $r_t \in \mathbb{R}$ , where we assume that reward is always bounded  $|r| \leq R \in \mathbb{R}^+$ . Given a sequence of observations  $\{\mathbf{x}_t\}$  and a sequence of rewards  $\{r_t\}$ , the long-term reward at  $t$  is defined as  $v_t = \sum_{k=t}^{\infty} \gamma^{k-t} r_s$ , where  $\gamma \in [0, 1)$  is a discount factor. Given a function space  $\mathcal{F}$  the learner chooses a predictor  $f$  at each time step from  $\mathcal{F}$  for predicting long-term rewards. Throughout this paper, we assume that any prediction made by a predictor  $f$  at a state  $\mathbf{x}$  is upper bounded as  $|f(\mathbf{x})| \leq P \in \mathbb{R}^+$ , for any  $f \in \mathcal{F}$  and  $\mathbf{x}$ .

At time step  $t = 0$ , the learner receives  $\mathbf{x}_0$ , initializes a predictor  $f_0 \in \mathcal{F}$  and makes prediction of  $v_0$  as  $f_0(\mathbf{x}_0)$ . Rounds of learning then proceeds as follows: the learner makes a prediction of  $v_t$  at step  $t$  as  $f_t(\mathbf{x}_t)$ ; the learner then observes a reward  $r_t$  and the next state  $\mathbf{x}_{t+1}$ ; the learner updates its predictor to  $f_{t+1}$ . This interaction repeats and is terminated after  $T$  steps. Throughout this paper, we call this problem setting as *online prediction of long-term reward*.

We define the *signed Bellman Error* at step  $t$  for predictor  $f_t$  as  $b_t = f_t(\mathbf{x}_t) - r_t - \gamma f_t(\mathbf{x}_{t+1})$ , which measures effectively how self consistent  $f_t$  is in its predictions between time step  $t$  and  $t + 1$ . For any  $f^* \in \mathcal{F}$ , we define the corresponding signed Bellman Error as  $b_t^* = f^*(\mathbf{x}_t) - r_t - \gamma f^*(\mathbf{x}_{t+1})$ . We denote the *Bellman Error* (BE) as the square of the signed Bellman error  $b_t^2$ .

The *Signed Prediction Error* of long-term reward at  $t$  for  $f_t$  is defined as  $e_t = f_t(\mathbf{x}_t) - v_t$  and  $e_t^* = f^*(\mathbf{x}_t) - v_t$  for  $f^*$  accordingly. We will typically be interested in bounding the *Prediction Error* (PE)  $e_t^2$  of a given algorithm in terms of the best possible PE. To lighten notation in the following sections, all sums over time indices implicitly run from 0 to  $T - 1$  unless explicitly noted otherwise.

### 2.2 NO-REGRET ONLINE LEARNING

Under our online setting, we will define loss functional  $l_t$  at step  $t$  as the traditional Bellman Error (BE):

$$l_t(f) = (f(\mathbf{x}_t) - r_t - \gamma f(\mathbf{x}_{t+1}))^2. \quad (1)$$

Note that  $l_t(f_t) = b_t^2$ .

Following the setting of online prediction of long-term reward, the learner computes predictor  $f_t$  at time step  $t$  and then receives the loss function  $l_t$  and the loss  $l_t(f_t)$  (after the learner receives  $r_t$  and  $\mathbf{x}_{t+1}$ ). We say that the online

algorithm is no-regret with respect to BE if:

$$\lim_{T \rightarrow \infty} \frac{1}{T} \sum l_t(f_t) - \frac{1}{T} \sum l_t(f^*) \leq 0, \quad (2)$$

for any predictor  $f^* \in \mathcal{F}$ , including the best predictor that minimizes  $\sum l_t(f)$  in hindsight.

The sequence of predictors  $f_t$  being no-regret intuitively means that the predictors are giving nearly as consistent predictions over time as is possible in that function class. One might wish that the sequence of predictors being no-regret is a sufficient condition for small prediction error. More formally, one might expect that if Eq. 2 holds for the sequence of predictors  $\{f_t\}$ ,  $\sum e_t^2$  can be upper bounded:

$$\lim_{T \rightarrow \infty} \frac{1}{T} \sum e_t^2 \leq C \frac{1}{T} \sum e_t^{*2}, \quad \forall f^* \in \mathcal{F}, \quad (3)$$

where  $C \in \mathbb{R}^+$  is a constant. Schapire and Warmuth (1996) showed such a conclusion (Eq. 3) for TD and later on Li (2008) proved such a conclusion for RG, both under the assumption that  $f(\mathbf{x})$  is linear.

Unfortunately, however, simply being no-regret (Eq. 2) is **not** a sufficient condition for upper bounding prediction error ( $\sum e_t^2$ ) as in the form of Eq. 3 for general function approximation form:

**Theorem 2.1** *There exists a sequence of  $\{f_t\}$  that is no-regret with respect to the loss functions  $\{l_t(f)\}$ , but no  $C \in \mathbb{R}^+$  exists that makes Eq. 3 hold.*

We prove Theorem 2.1 by providing an example in Appendix (see *Supplementary Material*) which is no-regret on  $\{l_t(f_t)\}$  (Eq. 2 holds) but Eq. 3 does not hold.

### 2.3 ONLINE STABILITY

The counter example that supports Theorem 2.1 presents a sequence of unstable predictors  $\{f_t\}$  where two successive predictors  $f_t$  and  $f_{t+1}$  vary wildly when predicting the long-term reward of  $\mathbf{x}_{t+1}$ . Such behavior is rather unusual for typical no-regret online learning algorithms. This suggests introducing a notion of *Online Stability* which we defined as:

**Definition Online Stability:** For the generated sequence of predictors  $f_t$ , we say the algorithm is online stable if:

$$\lim_{T \rightarrow \infty} \frac{1}{T} \sum (f_t(\mathbf{x}_{t+1}) - f_{t+1}(\mathbf{x}_{t+1}))^2 = 0. \quad (4)$$

Intuitively, the online stability means that on average the difference between successive predictors is eventually small. That is, the difference between  $f_t(\mathbf{x}_{t+1})$  and  $f_{t+1}(\mathbf{x}_{t+1})$  is small on average. Online stability is a general condition and does not severely limit the scope of the online learning algorithms. For instance, when  $f$  is linear,

the definition of stability of online learning in (Saha et al., 2012) (see Eq. 3 in Saha et al. (2012)) and (Ross and Bagnell, 2011) implies our form of online stability. We also show in the following section that many popular no-regret online learning algorithms including OGD, ONS and OWF, satisfy our online stability condition.

We show in next section that the sequence of predictors  $\{f_t\}$  being no-regret with respect to the loss functions  $\{l_t(f_t)\}$  **and satisfying the online stability condition** is sufficient for deriving an upper bound for prediction error as shown in Eq. 3.

## 3 ONLINE LEARNING FOR LONG-TERM REWARD PREDICTION

In this section, we combine the no-regret condition on loss functions  $\{l_t(f)\}$  and the online stability condition together to provide a worst-case analysis of sum of PE  $\sum e_t^2$ , which builds a connection between the PE of long-term rewards, regret and online stability.

More formally, our worst-case analysis shows that if the online algorithm running on the sequence of loss  $\{l_t(f)\}$  is no-regret and the generated sequence of predictors  $\{f_t\}$  satisfies the online stability condition, predictor error can be upper bounded in the form of Eq. 3. The analysis does not place any probabilistic assumption on the sequence of observations  $\{\mathbf{x}_t\}$  or any assumption on the form of predictors  $f \in \mathcal{F}$  (e.g.,  $f(\mathbf{x})$  does not have to be linear).

We start by first providing two important lemmas below:

**Lemma 3.1** *Let us define  $d_t = f_t(\mathbf{x}_t) - r_t - \gamma f_{t+1}(\mathbf{x}_{t+1})$ . We have:*

$$\sum d_t^2 \geq (1 - \gamma)^2 \sum e_t^2 + (\gamma^2 - \gamma)(e_T^2 - e_0^2). \quad (5)$$

Note that the difference between  $d_t$  and  $b_t$  is that  $d_t$  uses  $f_{t+1}(\mathbf{x}_{t+1})$  to estimate the long-term reward at step  $t + 1$  while  $b_t$  uses  $f_t(\mathbf{x}_{t+1})$ .

**Proof** Schapire and Warmuth (1996) implicitly showed that  $d_t = (f_t(\mathbf{x}) - v_t + v_t - (r_t + \gamma f_{t+1}(\mathbf{x}_{t+1}))) = (e_t - \gamma e_{t+1})$ . Squaring both sides and summing over from  $t = 0$  to  $t = T - 1$ , we get:

$$\begin{aligned} \sum d_t^2 &= \sum (e_t - \gamma e_{t+1})^2 \\ &= \sum e_t^2 + \gamma^2 \sum e_{t+1}^2 - 2\gamma \sum e_t e_{t+1} \\ &\geq \sum e_t^2 + \gamma^2 \sum e_{t+1}^2 - \gamma \sum e_t^2 - \gamma \sum e_{t+1}^2 \\ &= (1 - \gamma)^2 \sum e_t^2 + (\gamma^2 - \gamma)(e_T^2 - e_0^2). \end{aligned} \quad (6)$$

The first inequality is obtained by applying Young's inequality to  $2e_t e_{t+1}$  to get  $2e_t e_{t+1} \leq e_t^2 + e_{t+1}^2$ . ■



**Lemma 3.2** For any  $f^* \in \mathcal{F}$ , the prediction error  $\sum e_t^{*2}$  upper bounds the BE  $\sum b_t^{*2}$  as follows:

$$\sum b_t^{*2} \leq (1 + \gamma)^2 \sum e_t^{*2} + (\gamma + \gamma^2)(e_0^{*2} - e_T^{*2}). \quad (7)$$

The proof of Lemma 3.2 is similar to the one for Lemma 3.1. We present the proof in Appendix.

Now let us define a measure of the change in predictors between the steps of the online algorithm as  $\epsilon_t = f_t(\mathbf{x}_{t+1}) - f_{t+1}(\mathbf{x}_{t+1})$ , which is closely related to the online stability condition. The  $b_t$  and  $d_t$  are then closely related with each other by  $\epsilon_t$ :

$$\begin{aligned} d_t &= f_t(\mathbf{x}_t) - r_t - \gamma f_{t+1}(\mathbf{x}_{t+1}) - \gamma f_t(\mathbf{x}_{t+1}) + \gamma f_t(\mathbf{x}_{t+1}) \\ &= b_t + \gamma \epsilon_t. \end{aligned}$$

Squaring both sides, we get:

$$\begin{aligned} d_t^2 &= b_t^2 + 2b_t\gamma\epsilon_t + \gamma^2\epsilon_t^2 \leq b_t^2 + b_t^2 + \gamma^2\epsilon_t^2 + \gamma^2\epsilon_t^2 \\ &= 2b_t^2 + 2\gamma^2\epsilon_t^2, \end{aligned} \quad (8)$$

where the first inequality is coming from applying Young's inequality to  $2b_t\gamma\epsilon_t$  to get  $2b_t\gamma\epsilon_t \leq b_t^2 + \gamma^2\epsilon_t^2$ . We are now ready to state the following main theorem of this paper:

**Theorem 3.3** Assume a sequence of predictors  $\{f_t\}$  is generated by running some online algorithm on the sequence of loss functions  $\{l_t\}$ . For any predictor  $f^* \in \mathcal{F}$ , the sum of prediction errors  $\sum e_t^{*2}$  can be upper bounded as:

$$\begin{aligned} (1 - \gamma)^2 \sum e_t^{*2} &\leq 2 \sum (b_t^2 - b_t^{*2}) + 2\gamma^2 \sum \epsilon_t^2 \\ &\quad + 2(1 + \gamma)^2 \sum e_t^{*2} + M, \end{aligned} \quad (9)$$

where

$$M = 2(\gamma + \gamma^2)(e_0^{*2} - e_T^{*2}) - (\gamma^2 - \gamma)(e_T^2 - e_0^2).$$

By running a no-regret and online stable algorithm on the loss functions  $\{l_t(f)\}$ , as  $T \rightarrow \infty$ , the average prediction error is then asymptotically upper bounded by a constant factor of the best possible prediction error in the function class:

$$\lim_{T \rightarrow \infty} \frac{\sum e_t^{*2}}{T} \leq \frac{2(1 + \gamma)^2 \sum e_t^{*2}}{(1 - \gamma)^2 T}. \quad (10)$$

**Proof** Combining Lemma. 3.1 and Lemma. 3.2, we have:

$$\begin{aligned} &\sum d_t^2 - 2 \sum b_t^{*2} \\ &\geq (1 - \gamma)^2 \sum e_t^{*2} + (\gamma^2 - \gamma)(e_T^2 - e_0^2) \\ &\quad - 2(1 + \gamma)^2 \sum e_t^{*2} \\ &\quad - 2(\gamma + \gamma^2)(e_0^{*2} - e_T^{*2}). \end{aligned} \quad (11)$$

Subtracting  $2b_t^{*2}$  on both sides of Eq. 8, and then summing over from  $t = 1$  to  $T - 1$ , we have:

$$\sum d_t^2 - \sum 2b_t^{*2} \leq 2 \sum (b_t^2 - b_t^{*2}) + 2\gamma^2 \sum \epsilon_t^2.$$

Combining the above two inequalities together, we have:

$$\begin{aligned} &2 \sum (b_t^2 - b_t^{*2}) + 2\gamma^2 \sum \epsilon_t^2 \\ &\geq (1 - \gamma)^2 \sum e_t^{*2} + (\gamma^2 - \gamma)(e_T^2 - e_0^2) \\ &\quad - 2(1 + \gamma)^2 \sum e_t^{*2} - 2(\gamma + \gamma^2)(e_0^{*2} - e_T^{*2}). \end{aligned} \quad (12)$$

Rearrange inequality (12) and define  $M = 2(\gamma + \gamma^2)(e_0^{*2} - e_T^{*2}) - (\gamma^2 - \gamma)(e_T^2 - e_0^2)$ , we obtain inequality (9).

Assume that the  $\bar{f} = \arg \min_{f \in \mathcal{F}} \sum l_t(f)$ , then if the online algorithm is no-regret, we have

$$\begin{aligned} \frac{1}{T} \sum b_t^2 - b_t^{*2} &= \frac{1}{T} \sum l_t(f_t) - l_t(f^*) \\ &\leq \frac{1}{T} \sum l_t(f_t) - l_t(\bar{f}) \\ &= \frac{1}{T} \text{Regret} \leq 0, \quad T \rightarrow \infty. \end{aligned} \quad (13)$$

If the online algorithm satisfies the stability condition (Eq. 4), we have:  $\frac{1}{T} \sum \epsilon_t^2 = 0$  when  $T \rightarrow \infty$ .

Also, since we assume  $|f(\mathbf{x})| \leq P$  and  $|r| \leq R$ , we can see  $M$  must be upper bounded by some constant. Hence, we must have  $\frac{M}{T} = 0$ , as  $T \rightarrow \infty$ .

Under the conditions that the online algorithm is no-regret and satisfies online stability, we get Eq. 10 by dividing both sides of Eq. 9 by  $T$  and taking  $T$  to infinity. ■

Note that in Theorem 3.3, Eq. 9 holds for any  $f^* \in \mathcal{F}$ , including the  $f^*$  that minimizes the prediction error. But note that the one that minimizes prediction error, PE, does not necessarily optimize the BE, which may lead to an improvement of the bound in Eq. 10 in practice. To see this, note that we showed in the proof that for a no-regret algorithm:

$$\frac{1}{T} \sum b_t^2 - b_t^{*2} \leq \frac{1}{T} \text{Regret} \leq 0, \quad \text{as } T \rightarrow \infty. \quad (14)$$

Hence the limit of  $(1/T) \sum (b_t^2 - b_t^{*2})$  may be negative for some  $f^* \in \mathcal{F}$ , which could lead to a potential decrease in the upper bound of  $(1/T) \sum e_t^{*2}$  in Eq. 10 and give us a tighter bound in practice.

When  $e_t^* = 0, \forall t$ , from Theorem 3.3, it is easy to see that no-regret rate of  $(1/T) \sum (b_t^2 - b_t^{*2})$  and the online stability rate of  $(1/T) \sum \epsilon_t^2$  together determine the rate of the convergence of  $(1/T) \sum e_t^{*2}$ .

When  $T \rightarrow \infty$  and  $\gamma \rightarrow 1$  (specifically when  $\gamma \geq (1/\sqrt{2})$ ), our upper bound analysis in Eq. 10 is asymptotically tighter than the upper bound in Li (2008) (Eq. 12)

provided for  $\text{RG}$ , which is a special case of our approach as we demonstrate in the following section. As we will additionally show, a large number of popular no-regret online algorithms also satisfy the online stability condition, broadening the family of algorithms that can be used to learn predictors of long-term rewards.

## 4 ALGORITHMS

Our analysis in Sec. 3 provides a reduction from the online prediction of long-term reward to the no-regret online learning setting on a sequence of loss functions  $\{l_t(f)\}$  defined in Sec. 2.1, which enables us to develop a new set of algorithms. In this section, we give concrete examples of new Bellman Residual algorithms based on well-known no-regret online learning procedures such as Online Gradient Descent (OGD), Online Newton Step (ONS) and the Online variant of Frank Wolfe (OFW). The choice of algorithm depends on the size and sparsity level of features and the available computational budget. For instance, OGD generalizes  $\text{RG}$  and has  $O(n)$  computational complexity at every update step which makes it suitable for applications where sampling observations is cheap (e.g., RL for video games). ONS provides a logarithmic no-regret rate and could lead to faster convergence in practice, making it potentially suitable for applications where obtaining samples of observations is expensive (e.g., RL for a physical robot). Finally, OFW introduces sparsity and can be applied to problems where the feature dimension is larger than the number of samples.

Although the analysis in Sec. 3 does not place any assumption on predictors  $f \in \mathcal{F}$ , in practice to achieve the no-regret property on the loss functions  $\{l_t(f)\}$ , additional assumptions of loss functions (e.g., convexity) are needed. Since we discuss concrete no-regret online algorithms in this section, for  $\mathcal{F}$  we focus on vector spaces equipped with inner product. Specifically, we focus on two vector spaces: (1) *Reproducing Kernel Hilbert Spaces* (RKHS) where  $f = \sum \alpha_i K(\mathbf{x}_i, \cdot) \in \mathcal{F}$ , for some kernel  $K(\mathbf{x}, \cdot)$  and (2) spaces consisting of linear functions  $f(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$ ,  $\mathbf{w} \in \mathcal{W}^1$ . We now summarize the assumptions that we will use in section:

1. We assume  $\mathcal{F}$  (or  $\mathcal{W}$ ) is convex and bounded in a sense that the diameter of  $\mathcal{F}$  (or  $\mathcal{W}$ ) is upper bounded as  $\max_{f_1, f_2 \in \mathcal{F}} \|f_1 - f_2\| \leq D \in \mathbb{R}^+$ , where the norm  $\|f\|$  is defined by the inner product associated with the function space:  $\|f\|^2 = \langle f, f \rangle$ ;
2. We assume that  $\|\mathbf{x}_t\|_2 \leq X$ ,  $\forall t$ ,  $|K(\mathbf{x}_1, \mathbf{x}_2)| \leq K$ ,  $\forall \mathbf{x}_1, \mathbf{x}_2$ ,  $\|f\| \leq F$ ,  $\forall f \in \mathcal{F}$ , and  $\|\mathbf{w}\|_2 \leq W$ ,  $\forall \mathbf{w} \in \mathcal{W}$ , where  $K \in \mathbb{R}^+$ ,  $F \in \mathbb{R}^+$ ,  $W \in \mathbb{R}^+$ .

<sup>1</sup>Linear function space is a special case of RKHS. We discuss linear function separately since some online algorithms discussed here only work for linear functions

Any prediction  $f(\mathbf{x})$  is always bounded, since for RKHS,  $f(\mathbf{x})$  is bounded as  $|f(\mathbf{x})| \leq \|f\| \|K(\mathbf{x}, \cdot)\| \leq F\sqrt{K}$ , and for  $f(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$ , we also have  $|f(\mathbf{x})| \leq \|\mathbf{w}\|_2 \|\mathbf{x}\|_2 \leq WX$ . For notation simplicity, we then simply assume that  $f(\mathbf{x})$  is always bounded as  $|f(\mathbf{x})| \leq P$ ,  $P \in \mathbb{R}^+$ .

**Lemma 4.1** *With the above assumptions, for any pair of  $\mathbf{x}_t$  and  $\mathbf{x}_{t+1}$ , for RKHS, the loss functional  $l_t(f)$  is convex and Lipschitz continuous with respect to the norm defined by the inner product  $\langle \cdot, \cdot \rangle_K$ ; for  $f(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$ , the loss function  $l_t(\mathbf{w})$  is convex and Lipschitz continuous with respect to either  $L_1$  norm  $\|\cdot\|_1$  or  $L_2$  norm  $\|\cdot\|_2$ .*

We present the proof of the above lemma in Appendix.

### 4.1 GRADIENT-BASED APPROACHES

Before diving into the detailed examples of mirror descent and gradient based approaches, we first introduce an important lemma about the stability of one particular online algorithm: *Follow the Regularized Leader* (FTRL). It is well known that gradient-based and mirror descent approaches can be understood in the framework of FTRL. In particular, we only focus on the case where the loss functions are convex and  $L$ -Lipschitz continuous with a regularization that is strongly-convex. We refer reader to Shalev-Shwartz (2011) for detailed definitions of convex functions, Lipschitz continuous, and strong convexity.

The update rule of FTRL at step  $t$  can be summarized as:

$$f_{t+1} = \arg \min_{f \in \mathcal{F}} \sum_{i=0}^t l_i(f) + \frac{1}{\mu} R(f). \quad (15)$$

**Lemma 4.2** *For FTRL with convex and  $L$ -Lipschitz continuous loss functions  $l_t(f)$  and strongly convex regularization function  $R(f)$  (with respect to  $\|f\|$ ), we have:*

$$\sum \|f_t - f_{t+1}\| \leq LT\mu. \quad (16)$$

Setting  $\mu = \frac{1}{\sqrt{T}}$  to achieve no-regret property, then we have:

$$\lim_{T \rightarrow \infty} \frac{1}{T} \sum \|f_t - f_{t+1}\| = 0. \quad (17)$$

Similar proofs has been shown in (Ross and Bagnell, 2011) and (Saha et al., 2012). For completeness, we present the proof of the above lemma in Appendix following our notation and problem setting.

#### 4.1.1 Gradient Descent on BE

Gradient descent approaches can be understood in the FTRL framework where the convex loss functions in FTRL are replaced by a linear approximation:

$$f_{t+1} = \arg \min_{f \in \mathcal{F}} \sum_{i=0}^t \langle g_i, f \rangle + \frac{1}{\mu_t} R(f), \quad (18)$$

where  $g_t \in \partial l_t(f_t)$  is a sub-gradient of  $l_t$  at  $f_t$  and  $R(f)$  is a strongly convex regularizer.

We first consider the special case where  $f(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$  is linear. Note that our loss function is  $l_t(\mathbf{w}) = (\mathbf{w}^T \mathbf{x}_t - r_t - \gamma \mathbf{w}^T \mathbf{x}_{t+1})^2$  and its gradient  $g_t$  at  $\mathbf{w}_t$  is  $g_t = (\mathbf{w}_t^T \mathbf{x}_t - r_t - \gamma \mathbf{w}_t^T \mathbf{x}_{t+1})(\mathbf{x}_t - \gamma \mathbf{x}_{t+1})$ . Setting the regularization  $R(\mathbf{w}) = \frac{1}{2} \|\mathbf{w}\|_2^2$ , which is 1-strongly convex, we obtain the RG algorithm in Baird (1995), where the update step at  $t$  is:

$$\mathbf{w}_{t+1} := \mathbf{w}_t - \mu_t (\mathbf{w}_t^T (\mathbf{x}_t - \gamma \mathbf{x}_{t+1}) - r_t) (\mathbf{x}_t - \gamma \mathbf{x}_{t+1}),$$

Note that the linear loss function is convex and Lipschitz continuous ( $\|g_t\|_2$  is bounded based on our assumptions that  $\|\mathbf{x}\|_2$ ,  $|r|$  and  $\|\mathbf{w}\|_2$  are all bounded). Online Gradient Descent (OGD) is no-regret (Zinkevich, 2003) with  $\mu_t = 1/\sqrt{T}$  and from Lemma 4.2, we have:

$$\begin{aligned} & \frac{1}{T} \sum (\mathbf{w}_t^T \mathbf{x}_{t+1} - \mathbf{w}_{t+1}^T \mathbf{x}_{t+1})^2 \\ & \leq \frac{1}{T} \sum \|\mathbf{x}_{t+1}\|_2^2 \|\mathbf{w}_t - \mathbf{w}_{t+1}\|_2^2 \\ & \leq X^2 \frac{1}{T} \sum \|\mathbf{w}_t - \mathbf{w}_{t+1}\|_2^2 = 0, \quad T \rightarrow \infty, \end{aligned} \quad (19)$$

which exactly satisfies the online stability condition. Hence, RG enjoys the guarantee of our main theorem 3.3.

#### 4.1.2 Exponentiated Gradient Descent on BE

When we set the regularization  $R(\mathbf{w}) = \sum_i \mathbf{w}^i (\log(\mathbf{w}^i) - 1)$ , where for vector  $\mathbf{w}$ ,  $\mathbf{w}^i$  is the  $i$ -th component of  $\mathbf{w}$ , we generalize RG to *Exponentiated Gradient* (EG) descent as Precup and Sutton (1997) did for TD.

Since we assumed that  $\|\mathbf{w}\|_2 \leq W$ , then  $\|\mathbf{w}\|_1 \leq W'$  for  $W' \in \mathbb{R}^+$ . Then the regularization  $R(\mathbf{w})$  becomes  $(1/W')$ -strongly-convex and the loss function  $l_t(\mathbf{w})$  is Lipschitz continuous with respect to  $L_1$  norm  $\|\cdot\|_1$ . Solving Eq.18, we obtain the update step at  $t$  as:

$$\mathbf{w}_{t+1}^i = \mathbf{w}_t^i \exp(-\mu_t (\mathbf{w}_t^T (\mathbf{x}_t - \gamma \mathbf{x}_{t+1}) - r_t) (\mathbf{x}_t^i - \gamma \mathbf{x}_{t+1}^i)).$$

Similar to RG, using Lemma 4.2 (the norm in Lemma 4.2 becomes  $L_1$  norm) we can show that EG satisfies our online stability condition:

$$\begin{aligned} & \frac{1}{T} \sum (\mathbf{w}_t^T \mathbf{x}_{t+1} - \mathbf{w}_{t+1}^T \mathbf{x}_{t+1})^2 \\ & \leq \frac{1}{T} X^2 \sum \|\mathbf{w}_t - \mathbf{w}_{t+1}\|_1^2 \\ & \leq \frac{1}{T} X^2 \sum \|\mathbf{w}_t - \mathbf{w}_{t+1}\|_1^2 = 0, \quad T \rightarrow \infty, \end{aligned} \quad (20)$$

and is also no-regret on  $\{l_t(\mathbf{w})\}$  when  $\mu_t = 1/\sqrt{T}$ . Hence, EG descent approach enjoys our main theorem 3.3.

#### 4.1.3 Gradient Descent in RKHS

Now we consider functions  $f(\mathbf{x})$  that belongs to RKHS  $\mathcal{H}_{\mathcal{K}}$ . For the special case where  $R(f) = \frac{1}{2} \langle f, f \rangle_{\mathcal{K}}$ , we obtain an RG-style update based on functional gradient descent (Scholkopf and Smola, 2001):

$$\begin{aligned} f_{t+1} := & f_t - \mu_t \left( (f_t(\mathbf{x}_t) - r_t - \gamma f_t(\mathbf{x}_{t+1})) \right. \\ & \left. \times (K(\mathbf{x}_t, \cdot) - \gamma K(\mathbf{x}_{t+1}, \cdot)) \right). \end{aligned} \quad (21)$$

Similarly, it is straightforward to show that gradient descent in RKHS satisfies our stability condition and no-regret condition when  $\mu_t = 1/\sqrt{T}$ . Hence, gradient descent in RKHS also enjoys the predictive error guarantees derived in Theorem 3.3.

### 4.2 IMPLICIT ONLINE LEARNING

Recently Tamar et al. (2014) has demonstrated implicit online learning for temporal difference method. We consider the same approach for Bellman Residual minimization by applying implicit online learning from Kulis et al. (2010) to the loss functions  $\{l_t(f)\}$  and thus provide worst-case guarantees. Specifically at iteration  $t$ ,  $f_{t+1}$  is computed implicitly as:

$$\begin{aligned} f_{t+1} = & \arg \min_{f \in \mathcal{F}} \mathcal{D}_R(f, f_t) + \mu_t l_t(f) \\ = & \arg \min_{f \in \mathcal{F}} \mathcal{D}_R(f, f_t) + \mu_t (f(x_t) - \gamma f(x_{t+1}) - r_t)^2, \end{aligned} \quad (22)$$

where  $\mathcal{D}_R$  is a Bregman divergence. Saha et al. (2012) show that when  $\mu_t = 1/\sqrt{t}$ , the generating function  $R(f)$  is positive and strongly-convex, and the loss function  $l_t(f)$  is convex and Lipschitz continuous, implicit online learning is shown to be no-regret and also satisfies Eq. 17.

#### 4.2.1 Implicit Online Gradient Descent

Particularly, we first consider  $f = \sum \alpha_i K(\mathbf{x}_i, \cdot)$  in RKHS. Setting  $R(f) = \frac{1}{2} \langle f, f \rangle_{\mathcal{K}}$ , we have  $\mathcal{D}_R(f, f_t) = \frac{1}{2} \|f - f_t\|_{\mathcal{K}}^2$ . Then solving Eq. 22, we obtain the following update rule:

$$\begin{aligned} f_{t+1} := & f_t - \frac{\mu_t}{1 + \mu_t \|K(\mathbf{x}_t, \cdot) - \gamma K(\mathbf{x}_{t+1}, \cdot)\|_{\mathcal{K}}^2} \\ & \times (f_t(\mathbf{x}_t) - \gamma f_t(\mathbf{x}_{t+1}) - r_t) \\ & \times (K(\mathbf{x}_t, \cdot) - \gamma K(\mathbf{x}_{t+1}, \cdot)). \end{aligned}$$

When considering linear function  $f(x) = \mathbf{w}^T \mathbf{x}$  and  $R(\mathbf{w}) = \frac{1}{2} \|\mathbf{w}\|_2^2$ , we obtain a similar update step:

$$\begin{aligned} \mathbf{w}_{t+1} = & \mathbf{w}_t - \frac{\mu_t}{1 + \mu_t \|\mathbf{x}_t - \gamma \mathbf{x}_{t+1}\|_2^2} \\ & \times (\mathbf{w}_t^T (\mathbf{x}_t - \gamma \mathbf{x}_{t+1}) - r_t) (\mathbf{x}_t - \gamma \mathbf{x}_{t+1}). \end{aligned}$$

As we will show in the experiments, compared to RG (OGD on BE), implicit OGD on BE is less sensitive to the

choice of step-size, which enables us to set large step-size to achieve faster convergence for  $(1/T) \sum e_t^2$ . This phenomenon is also observed by Tamar et al. (2014) when they compare implicit temporal difference to the original TD algorithm (Sutton and Barto, 1998).

### 4.3 ONLINE NEWTON STEP

We also analyze an online second-order method: the Online Newton Step (ONS) (Hazan et al., 2006) for online prediction of long-term reward. For ONS, we only focus on linear function approximation  $f(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$ . We slightly adapt the ONS for our loss function  $l_t(\mathbf{w})$ . We first present the following lemma:

**Lemma 4.3** *For loss function  $l_t(\mathbf{w}) = (\mathbf{w}^T \mathbf{x}_t - r_t - \gamma \mathbf{w}^T \mathbf{x}_{t+1})^2$ , there exists a  $\lambda \in \mathbb{R}^+$ , such that for all  $\mathbf{w}$  and  $\mathbf{w}'$ :*

$$l_t(\mathbf{w}) \geq l_t(\mathbf{w}') + \nabla l_t(\mathbf{w}')^T (\mathbf{w} - \mathbf{w}') + \frac{\lambda}{2} (\mathbf{w} - \mathbf{w}')^T \nabla l_t(\mathbf{w}') \nabla l_t(\mathbf{w}')^T (\mathbf{w} - \mathbf{w}').$$

We present the proof of the above lemma in appendix.

With  $\lambda$ , then the iterative update rule for ONS is:

$$\mathbf{w}_t = \Pi_{\mathcal{W}}^{A_t} \left( \mathbf{w}_{t-1} - \frac{1}{\lambda} A_{t-1}^{-1} \nabla l_{t-1}(\mathbf{w}_{t-1}) \right), \quad (23)$$

where  $A_t = \sum_{i=0}^t \nabla l_t(\mathbf{w}_t) \nabla l_t(\mathbf{w}_t)^T + \epsilon I_n$ ,  $\epsilon \in \mathbb{R}^+$ , and  $\Pi_{\mathcal{W}}^{A_t}$  is a projection to  $\mathcal{W}$  with the norm induced by  $A_t$ :  $\Pi_{\mathcal{W}}^{A_t}(\mathbf{y}) = \arg \min_{\mathbf{w} \in \mathcal{W}} (\mathbf{w} - \mathbf{y})^T A_t (\mathbf{w} - \mathbf{y})$ , which makes this projection operator  $\Pi_{\mathcal{W}}^{A_t}$  not trivial and equal to solving a convex program usually.

Since  $\|\mathbf{x}\|_2 \leq X$ ,  $\|\mathbf{w}\|_2 \leq W$  and  $|r| \leq R$ , we have  $\|\nabla l_t(\mathbf{w})\|_2 \leq G$ , for  $G \in \mathbb{R}^+$ . The following lemma shows that ONS satisfies the our online stability condition:

**Lemma 4.4** *The sequence  $\{\mathbf{w}_t\}$  generated by ONS satisfies the online stability condition:*

$$\begin{aligned} & \frac{1}{T} \sum (\mathbf{w}_t^T \mathbf{x}_{t+1} - \mathbf{w}_{t+1}^T \mathbf{x}_{t+1})^2 \\ & \leq \frac{1}{T} \frac{X^2}{G^2 \lambda^2} n \log(T+1) = 0, \quad T \rightarrow \infty. \end{aligned} \quad (24)$$

The proof borrows ideas from Hazan et al. (2006) and is presented in the Appendix. Note that the convergence rate of  $\frac{1}{T} \sum (f_t(\mathbf{x}_{t+1}) - f_{t+1}(\mathbf{x}_{t+1}))^2$  is  $O(\log T/T)$ , which is the same as the no-regret rate of ONS.

### 4.4 PROJECTION-FREE ONLINE LEARNING

We analyze the Online Frank Wolfe (OFW) (Hazan and Kale, 2012) for online prediction of long-term reward. Previously introduced methods, including OGD, EG, OGD in

RKHS, Implicit OGD, and implicit OGD in RKHS, usually need a projection operation in each update step if the newly updated predictor is out of its pre-defined convex set  $\mathcal{F}$ , although in many cases, projection operations are simple<sup>2</sup>. OFW is a projection-free online method and every step involves solving a (typically very simple) linear programming problem. Again, we restrict our analysis to linear functions  $f = \mathbf{w}^T \mathbf{x}$  for OFW. Without loss of generality, we assume  $l_t(\mathbf{w})$  is  $L$ -Lipschitz continuous with some  $L \in \mathbb{R}^+$  (Lemma 4.1).

Applying the adversarial variant of OFW to the sequence of loss functions  $\{l_t(f)\}$ , we have the following iterative update step:

$$\begin{aligned} \mathbf{v}_t &= \arg \min_{\mathbf{w} \in \mathcal{W}} \nabla F_t(\mathbf{w}_t)^T \mathbf{w}, \\ \mathbf{w}_{t+1} &= (1 - t^{-\alpha}) \mathbf{w}_t + t^{-\alpha} \mathbf{v}_t, \end{aligned} \quad (25)$$

where  $\alpha = \frac{1}{4}$  and  $F_t(\mathbf{w})$  is computed as:

$$\begin{aligned} F_t(\mathbf{w}) &= \frac{1}{t+1} \sum_{i=0}^t \hat{l}_i(\mathbf{w}) \\ &= \frac{1}{t+1} \sum_{i=0}^t (\nabla l_i(\mathbf{w}_i)^T \mathbf{w} + \sigma_i \|\mathbf{w} - \mathbf{w}_0\|_2^2) \\ &= \frac{1}{t+1} \sum_{i=0}^t \left( (\mathbf{w}_i^T \mathbf{x}_i - r_i - \gamma \mathbf{w}_i^T \mathbf{x}_{i+1}) (\mathbf{x}_i - \gamma \mathbf{x}_{i+1})^T \mathbf{w} + \sigma_i \|\mathbf{w} - \mathbf{w}_0\|_2^2 \right), \end{aligned}$$

where  $\sigma_t = (L/D)t^{-1/4}$ ,  $\mathbf{w}_0$  is the initialization.

The online stability of OFW can be shown easily:

$$\begin{aligned} & \frac{1}{T} \sum (\mathbf{w}_t^T \mathbf{x}_{t+1} - \mathbf{w}_{t+1}^T \mathbf{x}_{t+1})^2 \\ & \leq \frac{1}{T} X^2 \sum \|\mathbf{w}_t - \mathbf{w}_{t+1}\|_2^2 \\ & \leq \frac{1}{T} X^2 \sum t^{-2\alpha} \|\mathbf{w}_t - \mathbf{v}_t\|_2^2 \\ & \leq \frac{1}{T} X^2 D^2 \sum t^{-2\alpha} = 0, \quad T \rightarrow \infty. \end{aligned}$$

The first inequality comes from Cauchy-Schwartz inequality and the assumption that  $\|\mathbf{x}\|_2 \leq X$ . The last equality follows from the fact that  $2\alpha > 0$ , and  $\frac{1}{T} \sum_{t=1}^T t^{-\xi} = 0$ , when  $\xi > 0$  and  $T \rightarrow \infty$ .

Note that the output of the OFW  $\mathbf{w}_t$  is sparse when  $\mathcal{W}$  is defined as  $\mathcal{W} = \{\mathbf{w} : \|\mathbf{w}\|_1 \leq W'\}$  for some  $W' \in \mathbb{R}^+$  and  $\mathbf{w}_0$  is initialized to the origin or any corner point of  $\mathcal{W}$ . This is because the output  $\mathbf{v}_t$  of Eq. 25 will be always one of the corner points of  $\mathcal{W}$  and  $\mathbf{w}_t$  hence is a linear combination of corner points  $\{\mathbf{w}_0, \mathbf{v}_0, \dots, \mathbf{v}_{t-1}\}$ , leading to the

<sup>2</sup>Usually, when  $\mathcal{W}$  is defined as  $\{\mathbf{w} : \|\mathbf{w}\|_2 \leq W\}$ , an  $L_2$  projection to such  $\mathcal{W}$  is easy and can be implemented in  $O(n)$ . The same for  $L_2$  projection in RKHS when  $\mathcal{F} = \{f : \|f\| \leq F\}$ .



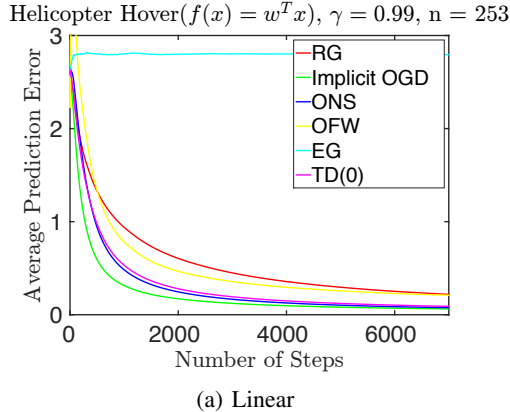


Figure 3: Convergence of prediction error for Helicopter Hover with linear function approximation

**Helicopter Hover** The helicopter simulator has a continuous 21-dimensional state space and a continuous 4-dimensional control. The reward is equal to the negative of the quadratic deviation to the targeted hover state. To generate sequence of states, we apply an LQR controller using linearized dynamics around the target hover state. We additionally corrupt the dynamics simulation with noise sampled from a Gaussian distribution. We used a degree-two polynomial feature that maps an original 21-dimensional state to a feature vector  $\mathbf{x} \in \mathbb{R}^{253}$  (Fig. 3(a)) and attempt to predict the long-term cost-to-go.

**Analysis of results** We fixed TD(0)’s step-size but a wide range of step-sizes were tried, and the best choice in terms of prediction error was used for TD(0). For RG, implicit OGD, and EG, we set the step-size to  $c/\sqrt{t}$ , where  $c$  is a constant. We also tried a range of  $c$  and chose the one that leads to the best performance. For all algorithms, we provided the same random initialization. All the results are computed by averaging over 100 random trials. As we can see from Fig. 3, ONS and implicit OGD give good convergence speed in general. Implicit OGD performed well with both RKHS and linear function approximation. Throughout the experiments, we found that implicit OGD was able to use a larger  $c$  to speed up convergence while still maintaining good stability. Surprisingly, our experimental results clearly show that our approaches have the possibility to achieve smaller prediction error than TD(0) (e.g., Fig. 2(b), bottom). This runs counter to the fact that the upper bound of prediction error provided by our analysis in Sec. 3 is looser than the upper bound of prediction error of TD(0) from both Li (2008) and Schapire and Warmuth (1996). Though our analysis is more general, further investigation is needed to tighten the worst-case bounds on our approach.

## 6 CONCLUSION

We established a general connection between the worst-case prediction of long term reward and Bellman errors for

stable prediction algorithms. We showed that together with this online stability condition, **any** no-regret online learning algorithm optimizing Bellman errors ensures small prediction errors. The stability condition is weak enough such that most popular no-regret online algorithms satisfy it. Our approach then suggests and provides soundness guarantees for online prediction of long-term reward using a broad new family of algorithms, including Online BE Newton Step, Online BE Frank Wolf, Implicit BE online learning (implicit gradient descent). The analysis itself can be applied to more general function space of hypotheses including Reproducing Kernel Hilbert Space representations and even to discrete hypothesis classes (i.e. trees). However we also want to point out that while our setting is very general, one might expect that in strongly non-Markovian situations there may fail to be a good predictor—e.g., no linear predictor using only the features of  $x_t$  can do a good job. In that sense our theorem in this paper is relative—essentially temporally coherent predictions (in the sense of small Bellman error) imply doing nearly as well as can be done at long term prediction: whether that is actually good performance depends on the quality of both features and hypothesis class, but not on any probabilistic assumptions.

## 7 DISCUSSION

Although our analysis provides broad and sound generalizations of RG, it does not provide guarantees on what we believe are the natural generalization of TD(0) or its variants as online algorithms on a sequence of temporal difference loss functions (TD-loss) which is defined as:

$$\tilde{l}_t(f) = (f(\mathbf{x}_t) - r_t - \gamma f_t(\mathbf{x}_{t+1}))^2. \quad (26)$$

Note that the difference between  $\tilde{l}_t$  and  $l_t$  is the subscript on the second predictor. The reason that we call it TD-loss is that when  $f(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$  is linear, applying OGD to  $\tilde{l}_t(\mathbf{w})$  with respect  $\mathbf{w}$  exactly reveals the update step of TD(0). By properly choosing step-size ( $\mu = O(1/\sqrt{T})$ ), OGD is no-regret on the TD-loss functions  $\{\tilde{l}_t(f)\}$ . Similar to our analysis of Bellman error algorithms, we believe that it is possible that no-regret property on TD-loss functions and stability condition of online algorithms together could lead us to similar predictive guarantees as shown in Theorem 3.3. In fact our empirical results (included in Appendix) suggested that such approaches are both sound and may outperform Bellman Residual methods. We leave it as future work to establish regret bounds for temporal difference minimizing online algorithms.

## 8 ACKNOWLEDGEMENTS

This work is supported by the ONR MURI grant N00014-09-1-1052, Reasoning in Reduced Information Spaces. We gratefully thank Arun Venkatraman for valuable discussions.

## References

- Leemon Baird. Residual algorithms: Reinforcement learning with function approximation. *Proceedings of the 12th International Conference on Machine Learning (ICML-95)*, pages 30–37, 1995.
- Adam Coates, Pieter Abbeel, and Andrew Y Ng. Learning for control from multiple demonstrations. *Proceedings of the 25th international conference on Machine learning - ICML '08*, pages 144–151, 2008.
- Yaakov Engel, Shie Mannor, and Ron Meir. Reinforcement learning with Gaussian processes. *Proceedings of the 22nd international conference on Machine learning*, 2005.
- Elad Hazan and Satyen Kale. Projection-free Online Learning. *29th International Conference on Machine Learning (ICML 2012)*, pages 521–528, 2012.
- Elad Hazan, Amit Agarwal, and Satyen Kale. Logarithmic regret algorithms for online convex optimization. *Proceedings of the 19th annual conference on Computational Learning Theory (COLT)*, pages 169–192, 2006.
- Brian Kulis, Peter L Bartlett, Bartlett Eecs, and Berkeley Edu. Implicit Online Learning. *Proceedings of the 27th international conference on Machine learning (ICML)*, pages 575–582, 2010.
- Lihong Li. A worst-case comparison between temporal difference and residual gradient with linear function approximation. *Proceedings of the 25th international conference on Machine learning - ICML '08*, pages 560–567, 2008.
- Doina Precup and Richard S. Sutton. Exponentiated Gradient Methods for Reinforcement Learning. In *Proceedings of the 14th International Conference on Machine Learning (ICML)*, 1997.
- M Robards, Peter Sunehag, Scott Sanner, and B Marthi. Sparse Kernel-SARSA( $\lambda$ ) with an Eligibility Trace. *ECML PKDD*, 2011.
- Stephane Ross and J. Andrew Bagnell. Stability Conditions for Online Learnability. *arXiv:1108.3154*, 2011.
- Ankan Saha, Prateek Jain, and Ambuj Tewari. The Interplay Between Stability and Regret in Online Learning. *arXiv preprint arXiv:1211.6158*, pages 1–19, 2012.
- Robert E. Schapire and Manfred K. Warmuth. On the worst-case analysis of temporal-difference learning algorithms. *Machine Learning*, 22(1):95–121, 1996. ISSN 0885-6125. doi: 10.1007/BF00114725.
- Bruno Scherrer. Should one compute the Temporal Difference fix point or minimize the Bellman Residual? The unified oblique projection view. *International Conference on Machine Learning (ICML 2010)*, 2010.
- Ralf Schoknecht and Artur Merke. TD(0) Converges Provably Faster than the Residual Gradient Algorithm. *International Conference on Machine Learning (ICML 2003)*, pages 680–687, 2003.
- Bernhard Scholkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press Cambridge, MA, USA, 2001.
- Shai Shalev-Shwartz. Online Learning and Online Convex Optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2011.
- Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.
- Aviv Tamar, Panos Toulis, Shie Mannor, and Edoardo M. Airoldi. Implicit Temporal Differences. *arXiv:1412.6734*, pages 1–6, 2014.
- Martin Zinkevich. Online Convex Programming and Generalized Infinitesimal Gradient Ascent. In *International Conference on Machine Learning (ICML 2003)*, pages 421–422, 2003.

---

# On the Error of Random Fourier Features

---

**Danica J. Sutherland**  
 Carnegie Mellon University  
 Pittsburgh, PA  
 dsuther1@cs.cmu.edu

**Jeff Schneider**  
 Carnegie Mellon University  
 Pittsburgh, PA  
 schneide@cs.cmu.edu

## Abstract

Kernel methods give powerful, flexible, and theoretically grounded approaches to solving many problems in machine learning. The standard approach, however, requires pairwise evaluations of a kernel function, which can lead to scalability issues for very large datasets. Rahimi and Recht (2007) suggested a popular approach to handling this problem, known as random Fourier features. The quality of this approximation, however, is not well understood. We improve the uniform error bound of that paper, as well as giving novel understandings of the embedding’s variance, approximation error, and use in some machine learning methods. We also point out that surprisingly, of the two main variants of those features, the more widely used is strictly higher-variance for the Gaussian kernel and has worse bounds.

## 1 INTRODUCTION

Kernel methods provide an elegant, theoretically well-founded, and powerful approach to solving many learning problems. Since traditional algorithms require the computation of a full  $N \times N$  pairwise kernel matrix to solve learning problems on  $N$  input instances, however, scaling these methods to large-scale datasets containing more than thousands of data points has proved challenging. Rahimi and Recht (2007) spurred interest in one very attractive approach: approximating a continuous shift-invariant kernel  $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$  by

$$k(x, y) \approx z(x)^\top z(y) =: s(x, y),$$

where  $z : \mathcal{X} \rightarrow \mathbb{R}^D$ . Then primal methods in  $\mathbb{R}^D$  can be used, allowing most learning problems to be solved in  $O(N)$  time (e.g. Joachims 2006). Recent work has also exploited these embeddings in some of the most-scalable kernel methods to date (Dai et al. 2014).

Rahimi and Recht (2007) give two such embeddings, based on the Fourier transform  $P(\omega)$  of the kernel  $k$ : one of the form

$$\tilde{z}(x) := \sqrt{\frac{2}{D}} \begin{bmatrix} \sin(\omega_1^\top x) \\ \cos(\omega_1^\top x) \\ \vdots \\ \sin(\omega_{D/2}^\top x) \\ \cos(\omega_{D/2}^\top x) \end{bmatrix}, \quad \omega_i \stackrel{iid}{\sim} P(\omega) \quad (1)$$

and another of the form

$$\check{z}(x) := \sqrt{\frac{2}{D}} \begin{bmatrix} \cos(\omega_1^\top x + b_1) \\ \vdots \\ \cos(\omega_{D/2}^\top x + b_{D/2}) \end{bmatrix}, \quad \begin{matrix} \omega_i \stackrel{iid}{\sim} P(\omega) \\ b_i \stackrel{iid}{\sim} \text{Unif}_{[0, 2\pi]} \end{matrix} \quad (2)$$

Bochner’s theorem (1959) guarantees that for any continuous positive-definite function  $k(x - y)$ , its Fourier transform will be a nonnegative measure; if  $k(0) = 1$ , it will be properly normalized. Letting  $\tilde{s}$  be the reconstruction based on  $\tilde{z}$  and  $\check{s}$  that for  $\check{z}$ , we have that:

$$\begin{aligned} \tilde{s}(x, y) &= \frac{1}{D/2} \sum_{i=1}^{D/2} \cos(\omega_i^\top (x - y)) \\ \check{s}(x, y) &= \frac{1}{D} \sum_{i=1}^{D/2} \cos(\omega_i^\top (x - y)) + \cos(\omega_i^\top (x + y) + 2b_i). \end{aligned}$$

Letting  $\Delta := x - y$ , we have:

$$\mathbb{E} \cos(\omega^\top \Delta) = \Re \int e^{\omega^\top \Delta \sqrt{-1}} dP(\omega) = \Re k(\Delta) \quad (3)$$

$$\mathbb{E}_\omega \mathbb{E}_b \cos(\omega^\top (x + y) + 2b) = 0. \quad (4)$$

Thus each  $s(x, y)$  is a mean of bounded terms with expectation  $k(x, y)$ . For a given embedding dimension  $D$ , it is not immediately obvious which approximation is preferable:  $\check{z}$  gives twice as many samples for  $\omega$ , but adds additional (non-shift-invariant) noise. The academic literature seems split on the issue: of the first 100 papers citing Rahimi and Recht (2007) in a Google Scholar search, 15 used either  $\check{z}$  or the equivalent complex formulation, 14



used  $\tilde{z}$ , 28 did not specify, and the remainder didn't use the embedding. (None discussed that there was a choice.) Not included in the count are Rahimi and Recht's later work (2008a; 2008b), which used  $\tilde{z}$ ; indeed, post-publication revisions of the original paper only discuss  $\tilde{z}$ . Practically, we are aware of three implementations in machine learning libraries, each of which use  $\tilde{z}$  at the time of writing: scikit-learn (Pedregosa et al. 2011), Shogun (Sonnenburg et al. 2010), and JSAT (Raff 2011-15).

We show that  $\tilde{z}$  is superior for the popular Gaussian kernel, as well as how to decide which to use for other kernels.

The primary previous analyses of these embeddings, outside the one in the original paper, have been by Rahimi and Recht (2008a), who bound the increase in error of empirical risk estimates when learning models in the induced RKHS, and by Yang et al. (2012), who compare the ability of the Nyström and Fourier embeddings to exploit eigengaps in the learning problem. We instead study the approximation directly, providing a complementary view of the quality of these embeddings.

Section 2.1 studies the variance of each embedding, showing that which is preferable depends on the kernel as well as the particular value of  $\Delta$ , but for the popular Gaussian kernel  $\tilde{s}$  is uniformly lower-variance. Section 2.2 studies uniform convergence bounds, tightening constants in the original  $\tilde{z}$  bound and proving a comparable one (with worse constants) for  $\tilde{z}$ , bounding the expectation of the maximal error, and providing exponential concentration about the mean. Section 2.3 studies the  $L_2$  convergence of each approximation;  $\tilde{z}$  is again superior for the Gaussian kernel. Section 3 discusses the effect of this approximation error when used in various machine learning methods. Section 4 evaluates the two embeddings and the bounds empirically.

## 2 APPROXIMATION ERROR

We will give various analyses of the error due to each approximation.

### 2.1 VARIANCE

(3) and (4) establish that  $\mathbb{E}s(\Delta) = k(\Delta)$ . What about the variance? We have that

$$\begin{aligned} & \text{Cov}(\tilde{s}(\Delta), \tilde{s}(\Delta')) \\ &= \text{Cov}\left(\frac{2}{D} \sum_{i=1}^{D/2} \cos(\omega_i^\top \Delta), \frac{2}{D} \sum_{i=1}^{D/2} \cos(\omega_i^\top \Delta')\right) \\ &= \frac{2}{D} \text{Cov}(\cos(\omega^\top \Delta), \cos(\omega^\top \Delta')) \\ &= \frac{2}{D} \left[\frac{1}{2}k(\Delta - \Delta') + \frac{1}{2}k(\Delta + \Delta') - k(\Delta)k(\Delta')\right] \end{aligned}$$

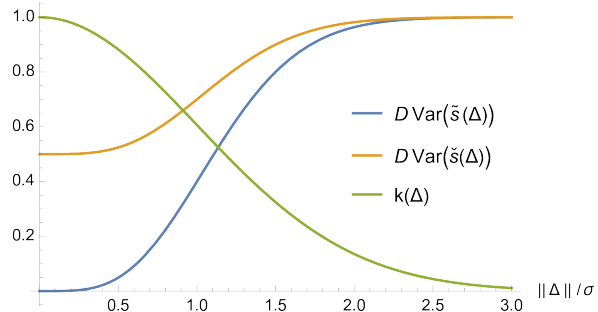


Figure 1: The variance per dimension of  $\tilde{s}$  (blue) and  $\check{s}$  (orange) for the Gaussian RBF kernel (green).

using  $\cos(\alpha)\cos(\beta) = \frac{1}{2}\cos(\alpha + \beta) + \frac{1}{2}\cos(\alpha - \beta)$  and also  $\mathbb{E}\cos(\omega^\top \Delta) = k(\Delta)$ . Thus

$$\text{Var} \tilde{s}(\Delta) = \frac{1}{D} [1 + k(2\Delta) - 2k(\Delta)^2]. \quad (5)$$

Similarly, denoting  $x + y$  by  $t$ ,

$$\begin{aligned} & \text{Cov}(\check{s}(x, y), \check{s}(x', y')) \\ &= \frac{1}{D} \text{Cov}(\cos(\omega^\top \Delta) + \cos(\omega^\top t + 2b), \\ & \quad \cos(\omega^\top \Delta') + \cos(\omega^\top t' + 2b)) \\ &= \frac{1}{D} \left[\frac{1}{2}k(\Delta - \Delta') + \frac{1}{2}k(\Delta + \Delta') - k(\Delta)k(\Delta') \right. \\ & \quad \left. + \frac{1}{2}k(t - t')\right] \end{aligned}$$

which gives

$$\text{Var} \check{s}(x, y) = \frac{1}{D} \left[1 + \frac{1}{2}k(2\Delta) - k(\Delta)^2\right]. \quad (6)$$

Thus  $\tilde{s}$  has lower variance than  $\check{s}$  if

$$\text{Var} \cos(\omega^\top \Delta) = \frac{1}{2} + \frac{1}{2}k(2\Delta) - k(\Delta)^2 \leq \frac{1}{2}. \quad (7)$$

The Gaussian kernel  $k(\Delta) = \exp\left(-\frac{\|\Delta\|^2}{2\sigma^2}\right)$  has

$$\text{Var} \cos(\omega^\top \Delta) = \frac{1}{2} \left(1 - \exp\left(-\frac{\|\Delta\|^2}{\sigma^2}\right)\right)^2 \leq \frac{1}{2},$$

so that  $\tilde{z}$  is always lower-variance than  $\check{z}$ , and the difference in variance is greatest when  $k(\Delta)$  is largest. This is illustrated in Figure 1.

### 2.2 UNIFORM ERROR BOUND

Let  $f(x, y) := s(x, y) - k(x, y)$  denote the error of the approximation. We will investigate  $\|f\|_\infty$ , i.e. the maximal approximation error across the domain of  $k$ . We first consider the bound given by Rahimi and Recht (2007), and then provide a new bound on  $\mathbb{E}\|f\|_\infty$  and its concentration around that mean.

### 2.2.1 Original High-Probability Bound

Claim 1 of Rahimi and Recht (2007) is that if  $\mathcal{X} \subset \mathbb{R}^d$  is compact with diameter  $\ell$ ,<sup>1</sup>

$$\Pr(\|f\|_\infty \geq \varepsilon) \leq 256 \left(\frac{\sigma_p \ell}{\varepsilon}\right)^2 \exp\left(-\frac{D\varepsilon^2}{8(d+2)}\right),$$

where  $\sigma_p^2 = \mathbb{E}[\omega^\top \omega] = \text{tr} \nabla^2 k(0)$  depends on the kernel.

It is not necessarily clear in that paper that this bound applies only to the  $\tilde{z}$  embedding; we can also tighten some constants. We first state the tightened bound for  $\tilde{z}$ .

**Proposition 1.** *Let  $k$  be a continuous shift-invariant positive-definite function  $k(x, y) = k(\Delta)$  defined on  $\mathcal{X} \subset \mathbb{R}^d$ , with  $k(0) = 1$  and such that  $\nabla^2 k(0)$  exists. Suppose  $\mathcal{X}$  is compact, with diameter  $\ell$ . Denote  $k$ 's Fourier transform as  $P(\omega)$ , which will be a probability distribution; let  $\sigma_p^2 = \mathbb{E}_p \|\omega\|^2$ . Let  $\tilde{z}$  be as in (1), and define  $\tilde{f}(x, y) := \tilde{z}(x)^\top \tilde{z}(y) - k(x, y)$ . For any  $\varepsilon > 0$ , let*

$$\alpha_\varepsilon := \min\left(1, \sup_{x, y \in \mathcal{X}} \frac{1}{2} + \frac{1}{2}k(2x, 2y) - k(x, y)^2 + \frac{1}{3}\varepsilon\right),$$

$$\beta_d := \left(\left(\frac{d}{2}\right)^{\frac{-d}{d+2}} + \left(\frac{d}{2}\right)^{\frac{2}{d+2}}\right) 2^{\frac{6d+2}{d+2}}.$$

Then, assuming only for the second statement that  $\varepsilon \leq \sigma_p \ell$ ,

$$\Pr(\|\tilde{f}\|_\infty \geq \varepsilon) \leq \beta_d \left(\frac{\sigma_p \ell}{\varepsilon}\right)^{\frac{2}{1+\frac{2}{d}}} \exp\left(-\frac{D\varepsilon^2}{8(d+2)\alpha_\varepsilon}\right)$$

$$\leq 66 \left(\frac{\sigma_p \ell}{\varepsilon}\right)^2 \exp\left(-\frac{D\varepsilon^2}{8(d+2)}\right).$$

Thus, we can achieve an embedding with pointwise error no more than  $\varepsilon$  with probability at least  $1 - \delta$  as long as

$$D \geq \frac{8(d+2)\alpha_\varepsilon}{\varepsilon^2} \left[\frac{2}{1+\frac{2}{d}} \log \frac{\sigma_p \ell}{\varepsilon} + \log \frac{\beta_d}{\delta}\right].$$

The proof strategy is very similar to that of Rahimi and Recht (2007): place an  $\varepsilon$ -net with radius  $r$  over  $\mathcal{X}_\Delta := \{x - y : x, y \in \mathcal{X}\}$ , bound the error  $\tilde{f}$  by  $\varepsilon/2$  at the centers of the net by Hoeffding's inequality (1963), and bound the Lipschitz constant of  $f$ , which is at most that of  $\tilde{s}$ , by  $\varepsilon/(2r)$  with Markov's inequality. The introduction of  $\alpha_\varepsilon$  is by replacing Hoeffding's inequality with that of Bernstein (1924) when it is tighter, using the variance from (5). The constant  $\beta_d$  is obtained by exactly optimizing the value of  $r$ , rather than the algebraically simpler value originally used;  $\beta_{64} = 66$  is its maximum, and  $\lim_{d \rightarrow \infty} \beta_d = 64$ , though it is lower for small  $d$ , as shown in Figure 2. The additional hypothesis, that  $\nabla^2 k(0)$  exists, is equivalent to the existence of the first two moments of  $P(\omega)$ ; a finite first moment is used in the proof, and of course without a finite second moment the bound is vacuous. The full proof is given in Appendix A.1.

<sup>1</sup>Note that our  $D$  is half of the  $D$  in Rahimi and Recht (2007), since we want to compare embeddings of the same dimension.

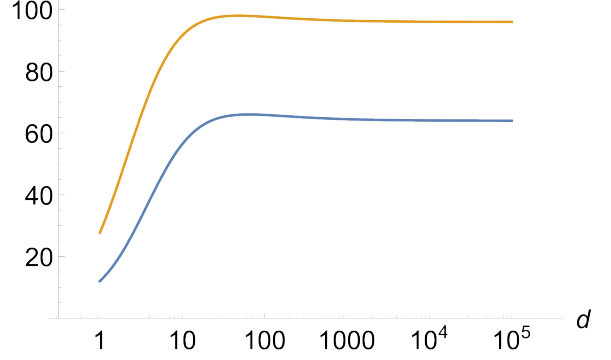


Figure 2: The coefficient  $\beta_d$  of Proposition 1 (blue, for  $\tilde{z}$ ) and  $\beta'_d$  of Proposition 2 (orange, for  $\check{z}$ ). Rahimi and Recht (2007) used a constant of 256 for  $\tilde{z}$ .

For the Gaussian kernel,  $\alpha_\varepsilon \leq \frac{1}{2} + \frac{1}{3}\varepsilon$  and  $\sigma_p^2 = d/\sigma^2$ ; the Bernstein bound is tighter when  $\varepsilon < \frac{3}{2}$ .

For  $\check{z}$ , since the embedding  $\check{s}$  is not shift-invariant, we must instead place the  $\varepsilon$ -net on  $\mathcal{X}^2$ . The additional noise in  $\check{s}$  also increases the expected Lipschitz constant and gives looser bounds on each term in the sum, though there are twice as many such terms. The corresponding bound is as follows:

**Proposition 2.** *Let  $k, \mathcal{X}, \ell, P(\omega)$ , and  $\sigma_p$  be as in Proposition 1. Define  $\check{z}$  by (2), and  $\check{f}(x, y) := \check{z}(x)^\top \check{z}(y) - k(x, y)$ . For any  $\varepsilon > 0$ , define*

$$\alpha'_\varepsilon := \min\left(1, \sup_{x, y \in \mathcal{X}} \frac{1}{4} + \frac{1}{8}k(2x, 2y) - \frac{1}{4}k(x, y)^2 + \frac{1}{6}\varepsilon\right),$$

$$\beta'_d := \left(d^{\frac{-d}{d+1}} + d^{\frac{1}{d+1}}\right) 2^{\frac{5d+1}{d+1}} 3^{\frac{d}{d+1}}.$$

Then, assuming only for the second statement that  $\varepsilon \leq \sigma_p \ell$ ,

$$\Pr(\|\check{f}\|_\infty \geq \varepsilon) \leq \beta'_d \left(\frac{\sigma_p \ell}{\varepsilon}\right)^{\frac{2}{1+\frac{1}{d}}} \exp\left(-\frac{D\varepsilon^2}{32(d+1)\alpha'_\varepsilon}\right)$$

$$\leq 98 \left(\frac{\sigma_p \ell}{\varepsilon}\right)^2 \exp\left(-\frac{D\varepsilon^2}{32(d+1)}\right).$$

Thus, we can achieve an embedding with pointwise error no more than  $\varepsilon$  with probability at least  $1 - \delta$  as long as

$$D \geq \frac{32(d+1)\alpha'_\varepsilon}{\varepsilon^2} \left[\frac{2}{1+\frac{1}{d}} \log \frac{\sigma_p \ell}{\varepsilon} + \log \frac{\beta'_d}{\delta}\right].$$

$\beta'_{48} = 98$ , and  $\lim_{d \rightarrow \infty} \beta'_d = 96$ , also shown in Figure 2. The full proof is given in Appendix A.2.

For the Gaussian kernel,  $\alpha'_\varepsilon \leq \frac{1}{4} + \frac{1}{6}\varepsilon$ , so that the Bernstein bound is essentially always superior.

### 2.2.2 Expected Max Error

Noting that  $\mathbb{E}\|f\|_\infty = \int_0^\infty \Pr(\|f\|_\infty \geq \varepsilon) d\varepsilon$ , one could consider bounding  $\mathbb{E}\|f\|_\infty$  via Propositions 1 and 2. Unfortunately, that integral diverges on  $(0, \gamma)$  for any  $\gamma > 0$ .

If we instead integrate the minimum of that bound and 1, the result depends on a solution to a transcendental equation, so analytical manipulation is difficult.

We can, however, use a slight generalization of Dudley’s entropy integral (1967) to obtain the following bound:

**Proposition 3.** *Let  $k$ ,  $\mathcal{X}$ ,  $\ell$ , and  $P(\omega)$  be as in Proposition 1. Define  $\tilde{z}$  by (1), and  $\tilde{f}(x, y) := \tilde{z}(x)^\top \tilde{z}(y) - k(x, y)$ . Let  $\mathcal{X}_\Delta := \{x - y \mid x, y \in \mathcal{X}\}$ ; suppose  $k$  is  $L$ -Lipschitz on  $\mathcal{X}_\Delta$ . Let  $R := \mathbb{E} \max_{i=1, \dots, \frac{D}{2}} \|\omega_i\|$ . Then*

$$\mathbb{E} \left[ \|\tilde{f}\|_\infty \right] \leq \frac{24\gamma\sqrt{d}\ell}{\sqrt{D}}(R + L)$$

where  $\gamma \approx 0.964$ .

The proof is given in Appendix A.3. In order to apply the method of Dudley (1967), we must work around  $\|\omega_i\|$  (which appears in the covariance of the error process) being potentially unbounded. To do so, we bound a process with truncated  $\|\omega_i\|$ , and then relate that bound to  $f$ .

For the Gaussian kernel,  $L = 1/(\sigma\sqrt{e})$  and<sup>2</sup>

$$\begin{aligned} R &\leq \left( \sqrt{2} \frac{\Gamma((d+1)/2)}{\Gamma(d/2)} + \sqrt{2 \log(D/2)} \right) / \sigma \\ &\leq \left( \sqrt{d} + \sqrt{2 \log(D/2)} \right) / \sigma. \end{aligned}$$

Thus  $\mathbb{E}\|\tilde{f}\|_\infty$  is less than

$$\frac{24\gamma\sqrt{d}\ell}{\sqrt{D}\sigma} \left( e^{-1/2} + \sqrt{d} + \sqrt{2 \log(D/2)} \right). \quad (8)$$

We can also prove an analogous bound for the  $\tilde{z}$  features:

**Proposition 4.** *Let  $k$ ,  $\mathcal{X}$ ,  $\ell$ , and  $P(\omega)$  be as in Proposition 1. Define  $\check{z}$  by (2), and  $\check{f}(x, y) := \check{z}(x)^\top \check{z}(y) - k(x, y)$ . Suppose  $k(\Delta)$  is  $L$ -Lipschitz. Let  $R := \mathbb{E} \max_{i=1, \dots, D} \|\omega_i\|$ . Then, for  $\mathcal{X}$  and  $D$  not extremely small,*

$$\mathbb{E} \left[ \|\check{f}\|_\infty \right] \leq \frac{48\gamma'_\mathcal{X}\ell\sqrt{d}}{\sqrt{D}}(R + L)$$

where  $0.803 < \gamma'_\mathcal{X} < 1.542$ . See Appendix A.4 for details on  $\gamma'_\mathcal{X}$  and the “not extremely small” assumption.

The proof is given in Appendix A.4. It is similar to that for Proposition 3, but the lack of shift invariance increases some constants and otherwise slightly complicates matters. Note also that the  $R$  of Proposition 4 is somewhat larger than that of Proposition 3.

<sup>2</sup>By the Gaussian concentration inequality (Boucheron et al. 2013, Theorem 5.6), each  $\|\omega\| - \mathbb{E}\|\omega\|$  is sub-Gaussian with variance factor  $\sigma^{-2}$ ; the claim follows from their Section 2.5.

### 2.2.3 Concentration About Mean

Bousquet’s inequality (2002) can be used to show exponential concentration of  $\sup f$  about its mean.

We consider  $\tilde{f}$  first. Let

$$\tilde{f}_\omega(\Delta) := \frac{2}{D} (\cos(\omega^\top \Delta) - k(\Delta)),$$

so  $f(\Delta) = \sum_{i=1}^{D/2} \tilde{f}_{\omega_i}(\Delta)$ . Define the “wimpy variance” of  $\tilde{f}/2$  (which we use so that  $|\tilde{f}/2| \leq 1$ ) as

$$\begin{aligned} \sigma_{\tilde{f}/2}^2 &:= \sup_{\Delta \in \mathcal{X}_\Delta} \sum_{i=1}^{D/2} \text{Var} \left[ \frac{1}{2} \tilde{f}_{\omega_i}(\Delta) \right] \\ &= \frac{1}{D} \sup_{\Delta \in \mathcal{X}_\Delta} [1 + k(2\Delta) - 2k(\Delta)^2] \\ &=: \frac{1}{D} \sigma_w^2, \end{aligned}$$

using (7). Clearly  $1 \leq \sigma_w^2 \leq 2$ ; for the Gaussian kernel, it is 1.

**Proposition 5.** *Let  $k$ ,  $\mathcal{X}$ , and  $P(\omega)$  be as in Proposition 1, and  $\tilde{z}$  be defined by (1). Let  $\tilde{f}(\Delta) = \tilde{z}(x)^\top \tilde{z}(y) - k(\Delta)$  for  $\Delta = x - y$ , and  $\sigma_w^2 := \sup_{\Delta \in \mathcal{X}_\Delta} 1 + k(2\Delta) - 2k(\Delta)^2$ . Then*

$$\begin{aligned} \Pr \left( \|\tilde{f}\|_\infty - \mathbb{E}\|\tilde{f}\|_\infty \geq \varepsilon \right) \\ \leq 2 \exp \left( - \frac{D\varepsilon^2}{D\mathbb{E}\|\tilde{f}\|_\infty + \frac{1}{2}\sigma_w^2 + \frac{D\varepsilon}{6}} \right). \end{aligned}$$

*Proof.* We use the Bernstein-style form of Theorem 12.5 of Boucheron et al. (2013) on  $\tilde{f}(\Delta)/2$  to obtain that  $\Pr \left( \sup \tilde{f} - \mathbb{E} \sup \tilde{f} \geq \varepsilon \right)$  is at most

$$\exp \left( - \frac{\varepsilon^2}{\mathbb{E} \sup \tilde{f} + \frac{1}{2}\sigma_{\tilde{f}/2}^2 + \frac{\varepsilon}{6}} \right).$$

The same holds for  $-\tilde{f}$ , and  $\mathbb{E} \sup \tilde{f} \leq \mathbb{E} \sup \|f\|_\infty$ ,  $\mathbb{E} \sup(-\tilde{f}) \leq \mathbb{E} \sup \|f\|_\infty$ . The claim follows by a union bound.  $\square$

A bound on the lower tail, unfortunately, is not available in the same form.

For  $\check{f}$ , note  $|\check{f}| \leq 3$ , so we use  $\check{f}/3$ . Letting  $\check{f}_\omega := \frac{1}{D} (\cos(\omega^\top \Delta) - k(\Delta))$ , we have  $\sigma_{\check{f}/3}^2 = \frac{1}{18D} (\sigma_w^2 + 1)$ . Thus the same argument gives us:

**Proposition 6.** *Let  $k$  and  $\mathcal{X}$  be as in Proposition 1, with  $P(\omega)$  defined as there. Let  $\check{z}$  be as in (2),  $\check{f}(x, y) = \check{z}(x)^\top \check{z}(y) - k(x, y)$ , and define  $\sigma_w$  as above. Then*

$$\begin{aligned} \Pr \left( \|\check{f}\|_\infty - \mathbb{E}\|\check{f}\|_\infty \geq \varepsilon \right) \\ \leq 2 \exp \left( - \frac{D\varepsilon^2}{\frac{4}{9}D\mathbb{E}\|\check{f}\|_\infty + \frac{1}{81}(\sigma_w^2 + 1) + \frac{2}{27}D\varepsilon} \right). \end{aligned}$$

Note that Proposition 6 actually gives a somewhat tighter concentration than Proposition 5. This is most likely because, between the space of possible errors being larger and the higher base variance illustrated in Figure 1, the  $\check{f}$  error function has more “opportunities” to achieve its maximal error. The experimental results (Figure 5) show that, at least in one case,  $\|\check{f}\|_\infty$  does concentrate about its mean more tightly, but that mean is enough higher than that of  $\|\tilde{f}\|_\infty$  that  $\|\check{f}\|_\infty$  stochastically dominates  $\|\tilde{f}\|_\infty$ .

### 2.3 $L_2$ ERROR BOUND

$L_\infty$  bounds provide useful guarantees, but are very strict. It can also be useful to consider a less stringent error measure. Let  $\mu$  be a  $\sigma$ -finite measure on  $\mathcal{X} \times \mathcal{X}$ ; define

$$\|f\|_\mu^2 := \int_{\mathcal{X}^2} f(x, y)^2 d\mu(x, y). \quad (9)$$

First, we have that

$$\begin{aligned} \mathbb{E}\|\tilde{f}\|_\mu^2 &= \mathbb{E} \int_{\mathcal{X}^2} \tilde{f}(x, y)^2 d\mu(x, y) \\ &= \int_{\mathcal{X}^2} \mathbb{E} \tilde{f}(x, y)^2 d\mu(x, y) \\ &= \int_{\mathcal{X}^2} \frac{1}{D} [1 + k(2x, 2y) - 2k(x, y)^2] d\mu(x, y) \\ &= \frac{1}{D} \left[ \mu(\mathcal{X}^2) + \int_{\mathcal{X}^2} k(2x, 2y) d\mu(x, y) - 2\|k\|_\mu^2 \right] \\ \mathbb{E}\|\check{f}\|_\mu^2 &= \frac{1}{D} \left[ \mu(\mathcal{X}^2) + \frac{1}{2} \int_{\mathcal{X}^2} k(2x, 2y) d\mu(x, y) - \|k\|_\mu^2 \right] \end{aligned} \quad (10)$$

where (10) is justified by Tonelli’s theorem.

If  $\mu = P_X \times P_Y$  is a joint distribution of independent variables, then  $\int_{\mathcal{X}^2} k(2x, 2y) d\mu(x, y) = \text{MMK}(P_{2X}, P_{2Y})$ , where MMK is the mean map kernel (see Section 3.3). Likewise,  $\|k\|_\mu^2 = \text{MMK}(P_X, P_Y)$  using the kernel  $k^2$ .<sup>3</sup>

Viewing  $\|\tilde{f}\|_\mu$  as a function of  $\omega_1, \dots, \omega_{D/2}$ , changing  $\omega_i$  to a different  $\hat{\omega}_i$  changes the value of  $\|\tilde{f}\|_\mu$  by at most  $4\frac{D+1}{D^2}\mu(\mathcal{X}^2)$ ; this can be seen by simple algebra and is shown in Appendix B.1. Thus McDiarmid (1989) gives us an exponential concentration bound:

**Proposition 7.** *Let  $k$  be a continuous shift-invariant positive-definite function  $k(x, y) = k(\Delta)$  defined on  $\mathcal{X} \subseteq \mathbb{R}^d$ , with  $k(0) = 1$ . Let  $\mu$  be a  $\sigma$ -finite measure on  $\mathcal{X}^2$ , and define  $\|\cdot\|_\mu^2$  as in (9). Define  $\tilde{z}$  as in (1) and let  $\tilde{f}(x, y) = \tilde{z}(x)^\top \tilde{z}(y) - k(x, y)$ . Let  $\mathcal{M} := \mu(\mathcal{X}^2)$ . Then*

$$\begin{aligned} \Pr \left( \left| \|\tilde{f}\|_\mu^2 - \mathbb{E}\|\tilde{f}\|_\mu^2 \right| \geq \varepsilon \right) &\leq 2 \exp \left( \frac{-D^3 \varepsilon^2}{8(4D+1)^2 \mathcal{M}^2} \right) \\ &\leq 2 \exp \left( \frac{-D \varepsilon^2}{200 \mathcal{M}^2} \right). \end{aligned}$$

<sup>3</sup> $k^2$  is also a PSD kernel, by the Schur product theorem.

The second version of the bound is simpler, but somewhat looser for  $D \gg 1$ ; asymptotically, the coefficient of the denominator becomes 128.

Similarly, the variation of  $\|\check{f}\|_\mu$  is bounded by at most  $32\frac{D+1}{D^2}\mu(\mathcal{X}^2)$  (shown in Appendix B.2). Thus:

**Proposition 8.** *Let  $k$ ,  $\mu$ ,  $\|\cdot\|_\mu$ , and  $\mathcal{M}$  be as in Proposition 7. Define  $\check{z}$  as in (2) and let  $\check{f}(x, y) = \check{z}(x)^\top \check{z}(y) - k(x, y)$ . Then*

$$\begin{aligned} \Pr \left( \left| \|\check{f}\|_\mu^2 - \mathbb{E}\|\check{f}\|_\mu^2 \right| \geq \varepsilon \right) &\leq 2 \exp \left( \frac{-D^3 \varepsilon^2}{512(D+1)^2 \mathcal{M}^2} \right) \\ &\leq 2 \exp \left( \frac{-D \varepsilon^2}{2048 \mathcal{M}^2} \right). \end{aligned}$$

The cost of a simpler dependence on  $D$  is higher here; the asymptotic coefficient of the denominator is 512.

## 3 DOWNSTREAM ERROR

Rahimi and Recht (2008a; 2008b) give a bound on the  $L_2$  distance between any given function in the reproducing kernel Hilbert space (RKHS) induced by  $k$  and the closest function in the RKHS of  $s$ : results invaluable for the study of learning rates. In some situations, however, it is useful to consider not the learning-theoretic convergence of hypotheses to the assumed “true” function, but rather directly consider the difference in predictions due to using the  $z$  embedding instead of the exact kernel  $k$ .

### 3.1 KERNEL RIDGE REGRESSION

We first consider kernel ridge regression (KRR; Saunders et al. 1998). Suppose we are given  $n$  training pairs  $(x_i, y_i) \in \mathbb{R}^d \times \mathbb{R}$  as well as a regularization parameter  $\lambda = n\lambda_0 > 0$ . We construct the training Gram matrix  $K$  by  $K_{ij} = k(x_i, x_j)$ . KRR gives predictions  $h(x) = \alpha^\top k_x$ , where  $\alpha = (K + \lambda I)^{-1} y$  and  $k_x$  is the vector with  $i$ th component  $k(x_i, x)$ .<sup>4</sup> When using Fourier features, one would not use  $\alpha$ , but instead a primal weight vector  $w$ ; still, it will be useful for us to analyze the situation in the dual.

Proposition 1 of Cortes et al. (2010) bounds the change in KRR predictions from approximating the kernel matrix  $K$  by  $\hat{K}$ , in terms of  $\|\hat{K} - K\|_2$ . They assume, however, that the kernel evaluations at test time  $k_x$  are unapproximated, which is certainly not the case when using Fourier features. We therefore extend their result to Proposition 9 before using it to analyze the performance of Fourier features.

<sup>4</sup>If a bias term is desired, we can use  $k'(x, x') = k(x, x') + 1$  by appending a constant feature 1 to the embedding  $z$ . Because this change is accounted for exactly, it affects the error analysis here only in that we must use  $\sup |k(x, y)| \leq 2$ , in which case the first factor of (11) becomes  $(\lambda_0 + 2)/\lambda_0^2$ .

**Proposition 9.** Given a training set  $\{(x_i, y_i)\}_{i=1}^n$ , with  $x_i \in \mathbb{R}^d$  and  $y_i \in \mathbb{R}$ , let  $h(x)$  denote the result of kernel ridge regression using the PSD training kernel matrix  $K$  and test kernel values  $k_x$ . Let  $\hat{h}(x)$  be the same using a PSD approximation to the training kernel matrix  $\hat{K}$  and test kernel values  $\hat{k}_x$ . Further assume that the training labels are centered,  $\sum_{i=1}^n y_i = 0$ , and let  $\sigma_y^2 := \frac{1}{n} \sum_{i=1}^n y_i^2$ . Also suppose  $\|k_x\|_\infty \leq \kappa$ . Then:

$$|h'(x) - h(x)| \leq \frac{\sigma_y}{\sqrt{n}\lambda_0} \|\hat{k}_x - k_x\| + \frac{\kappa\sigma_y}{n\lambda_0^2} \|\hat{K} - K\|_2.$$

*Proof.* Let  $\alpha = (K + \lambda I)^{-1}y$ ,  $\hat{\alpha} = (\hat{K} + \lambda I)^{-1}y$ . Thus, using  $\hat{M}^{-1} - M^{-1} = -\hat{M}^{-1}(\hat{M} - M)M^{-1}$ , we have

$$\begin{aligned} \hat{\alpha} - \alpha &= -(\hat{K} + \lambda I)^{-1}(\hat{K} - K)(K + \lambda I)^{-1}y \\ \|\hat{\alpha} - \alpha\| &\leq \|(\hat{K} + \lambda I)^{-1}\|_2 \|\hat{K} - K\|_2 \|(K + \lambda I)^{-1}\|_2 \|y\| \\ &\leq \frac{1}{\lambda^2} \|\hat{K} - K\|_2 \|y\| \end{aligned}$$

since the smallest eigenvalues of  $K + \lambda I$  and  $\hat{K} + \lambda I$  are at least  $\lambda$ . Since  $\|k_x\| \leq \sqrt{n}\kappa$  and  $\|\hat{\alpha}\| \leq \|y\|/\lambda$ :

$$\begin{aligned} |\hat{h}(x) - h(x)| &= |\hat{\alpha}^\top \hat{k}_x - \alpha^\top k_x| \\ &= |\hat{\alpha}^\top (\hat{k}_x - k_x) + (\hat{\alpha} - \alpha)^\top k_x| \\ &\leq \|\hat{\alpha}\| \|\hat{k}_x - k_x\| + \|\hat{\alpha} - \alpha\| \|k_x\| \\ &\leq \frac{\|y\|}{\lambda} \|\hat{k}_x - k_x\| + \frac{\sqrt{n}\kappa\|y\|}{\lambda^2} \|\hat{K} - K\|_2. \end{aligned}$$

The claim follows from  $\lambda = n\lambda_0$ ,  $\|y\| = \sqrt{n}\sigma_y$ .  $\square$

Suppose that, per the uniform error bounds of Section 2.2,  $\sup |k(x, y) - s(x, y)| \leq \varepsilon$ . Then  $\|\hat{k}_x - k_x\| \leq \sqrt{n}\varepsilon$  and  $\|\hat{K} - K\|_2 \leq \|\hat{K} - K\|_F \leq n\varepsilon$ , and Proposition 9 gives

$$\begin{aligned} |\hat{h}(x) - h(x)| &\leq \frac{\sigma_y}{\sqrt{n}\lambda_0} \sqrt{n}\varepsilon + \frac{\sigma_y}{n\lambda_0^2} n\varepsilon \\ &\leq \frac{\lambda_0 + 1}{\lambda_0^2} \sigma_y \varepsilon. \end{aligned} \quad (11)$$

Thus

$$\Pr(|h'(x) - h(x)| \geq \varepsilon) \leq \Pr\left(\|f\|_\infty \geq \frac{\lambda_0^2 \varepsilon}{(\lambda_0 + 1)\sigma_y}\right).$$

which we can bound with Proposition 1 or 2. We can therefore guarantee  $|h(x) - h'(x)| \leq \varepsilon$  with probability at least  $\delta$  if

$$D = \Omega\left(d \left(\frac{(\lambda_0 + 1)\sigma_y}{\lambda_0^2 \varepsilon}\right)^2 \left[\log \delta + \log \frac{\lambda_0^2 \varepsilon}{(\lambda_0 + 1)\sigma_y} - \log \sigma_p \ell\right]\right).$$

Note that this rate does not depend on  $n$ . If we want  $h'(x) \rightarrow h(x)$  at least as fast as  $h(x)$ 's convergence rate of  $O(1/\sqrt{n})$  (Bousquet and Elisseeff 2001), ignoring the logarithmic terms, we thus need  $D$  to be linear in  $n$ , matching the conclusion of Rahimi and Recht (2008a).

## 3.2 SUPPORT VECTOR MACHINES

Consider a Support Vector Machine (SVM) classifier with no offset, such that  $h(x) = w^\top \Phi(x)$  for a kernel embedding  $\Phi(x) : \mathcal{X} \rightarrow \mathcal{H}$  and  $w$  is found by

$$\operatorname{argmin}_{w \in \mathcal{H}} \frac{1}{2} \|w\|^2 + \frac{C_0}{n} \sum_{i=1}^n \max(0, 1 - y_i \langle w, \Phi(x_i) \rangle)$$

where  $\{(x_i, y_i)\}_{i=1}^n$  is our training set with  $y_i \in \{-1, 1\}$ , and the decision function is  $h(x) = \langle w, \Phi(x) \rangle$ .<sup>5</sup> For a given  $x$ , Cortes et al. (2010) consider an embedding in  $\mathcal{H} = \mathbb{R}^{n+1}$  which is equivalent on the given set of points. They bound  $|\hat{h}(x) - h(x)|$  in terms of  $\|\hat{K} - K\|_2$  in their Proposition 2, but again assume that the test-time kernel values  $k_x$  are exact. We will again extend their result in Proposition 10:

**Proposition 10.** Given a training set  $\{(x_i, y_i)\}_{i=1}^n$ , with  $x_i \in \mathbb{R}^d$  and  $y_i \in \{-1, 1\}$ , let  $h(x)$  denote the decision function of an SVM classifier using the PSD training matrix  $K$  and test kernel values  $k_x$ . Let  $\hat{h}(x)$  be the same using a PSD approximation to the training kernel matrix  $\hat{K}$  and test kernel values  $\hat{k}_x$ . Suppose  $\sup k(x, x) \leq \kappa$ . Then:

$$\begin{aligned} |\hat{h}(x) - h(x)| &\leq \sqrt{2}\kappa^{\frac{3}{4}} C_0 \left(\|\hat{K} - K\|_2 + \|\hat{k}_x - k_x\| + |f_x|\right)^{1/4} \\ &\quad + \sqrt{\kappa} C_0 \left(\|\hat{K} - K\|_2 + \|\hat{k}_x - k_x\| + |f_x|\right)^{1/2}, \end{aligned}$$

where  $f_x = \hat{k}(x, x) - k(x, x)$ .

*Proof.* Use the setup of Section 2.2 of Cortes et al. (2010). In particular, we will use  $\|w\| \leq \sqrt{\kappa} C_0$  and their (16-17):

$$\begin{aligned} \Phi(x_i) &= K_x^{1/2} e_i \\ \|\hat{w} - w\|^2 &\leq 2C_0^2 \sqrt{\kappa} \|\hat{K}_x^{1/2} - K_x^{1/2}\|, \end{aligned}$$

where  $K_x = \begin{bmatrix} K & k_x \\ k_x^\top & k(x, x) \end{bmatrix}$  and  $e_i$  the  $i$ th standard basis.

Further, Lemma 1 of Cortes et al. (2010) says that  $\|\hat{K}_x^{1/2} - K_x^{1/2}\|_2 \leq \|\hat{K}_x - K_x\|_2^{1/2}$ . Let  $f_x := \hat{k}(x, x) - k(x, x)$ ; Then, by Weyl's inequality for singular values,

$$\left\| \begin{bmatrix} \hat{K} - K & \hat{k}_x - k_x \\ \hat{k}_x^\top - k_x^\top & f_x \end{bmatrix} \right\|_2 \leq \|\hat{K} - K\|_2 + \|\hat{k}_x - k_x\| + |f_x|.$$

<sup>5</sup>We again assume there is no bias term for simplicity; adding a constant feature again changes the analysis only in that it makes the  $\kappa$  of Proposition 10 2 instead of 1.

Thus

$$\begin{aligned}
& |\hat{h}(x) - h(x)| \\
&= \left| (\hat{w} - w)^\top \hat{\Phi}(x) + w^\top (\hat{\Phi}(x) - \Phi(x)) \right| \\
&\leq \|\hat{w} - w\| \|\hat{\Phi}(x)\| + \|w\| \|\hat{\Phi}(x) - \Phi(x)\| \\
&\leq \sqrt{2}\kappa^{\frac{1}{4}} C_0 \|\hat{K}_x^{1/2} - K_x^{1/2}\|_2^{1/2} \sqrt{\kappa} \\
&\quad + \sqrt{\kappa} C_0 \|(\hat{K}_x^{1/2} - K_x^{1/2})e_{n+1}\| \\
&\leq \sqrt{2}\kappa^{\frac{3}{4}} C_0 \|\hat{K}_x - K_x\|_2^{1/4} \\
&\quad + \sqrt{\kappa} C_0 \|\hat{K}_x - K_x\|_2^{1/2} \\
&\leq \sqrt{2}\kappa^{\frac{3}{4}} C_0 \left( \|\hat{K} - K\|_2 + \|\hat{k}_x - k_x\| + |f_x| \right)^{1/4} \\
&\quad + \sqrt{\kappa} C_0 \left( \|\hat{K} - K\|_2 + \|\hat{k}_x - k_x\| + |f_x| \right)^{1/2}
\end{aligned}$$

as claimed.  $\square$

Suppose that  $\sup|k(x, y) - s(x, y)| \leq \varepsilon$ . Then, as in the last section,  $\|\hat{k}_x - k_x\| \leq \sqrt{n}\varepsilon$  and  $\|\hat{K} - K\|_2 \leq n\varepsilon$ . Then, letting  $\gamma$  be 0 for  $\tilde{z}$  and 1 for  $\check{z}$ , Proposition 10 gives

$$\begin{aligned}
|\hat{h}(x) - h(x)| &\leq \sqrt{2}C_0 (n + \sqrt{n} + \gamma)^{1/4} \varepsilon^{1/4} \\
&\quad + C_0 (n + \sqrt{n} + \gamma)^{1/2} \varepsilon^{1/2}.
\end{aligned}$$

Then  $|\hat{h}(x) - h(x)| \geq u$  only if

$$\varepsilon \leq \frac{2C_0^2 + 4C_0u + u^2 - 2(C_0 + u)\sqrt{C_0(C_0 + 2u)}}{C_0^2(n + \sqrt{n} + \gamma)}.$$

This bound has the unfortunate property of requiring the approximation to be *more* accurate as the training set size increases, and thus can prove only a very loose upper bound on the number of features needed to achieve a given approximation accuracy, due to the looseness of Proposition 10. Analyses of generalization error in the induced RKHS, such as Rahimi and Recht (2008a) and Yang et al. (2012), are more useful in this case.

### 3.3 MAXIMUM MEAN DISCREPANCY

Another area of application for random Fourier embeddings is to the mean embedding of distributions, which uses some kernel  $k$  to represent a probability distribution  $P$  in the RKHS induced by  $k$  as  $\varphi(P) = \mathbb{E}_{x \sim P} [k(x, \cdot)]$ . For samples  $\{X_i\}_{i=1}^n \sim P$  and  $\{Y_j\}_{j=1}^m \sim Q$ , we can estimate the inner product in the embedding space, the *mean map kernel* (MMK), by

$$\text{MMK}(X, Y) := \frac{1}{nm} \sum_{i=1}^n \sum_{j=1}^m k(X_i, Y_j) \approx \langle \varphi(P), \varphi(Q) \rangle.$$

The distance  $\|\varphi(P) - \varphi(Q)\|$  is known as the *maximum mean discrepancy* (MMD), which can be estimated with:

$$\begin{aligned}
& \|\varphi(P) - \varphi(Q)\|^2 \\
&= \langle \varphi(P), \varphi(P) \rangle + \langle \varphi(Q), \varphi(Q) \rangle - 2 \langle \varphi(P), \varphi(Q) \rangle.
\end{aligned}$$

MMK( $X, X$ ) is a biased estimator, because of the  $k(X_i, X_i)$  and  $k(Y_i, Y_i)$  terms; removing them gives an unbiased estimator (Gretton et al. 2012). The MMK can be used in standard kernel methods to perform learning on probability distributions, such as when images are treated as sets of local patch descriptors (Muandet et al. 2012) or documents as sets of word descriptors (Yoshikawa et al. 2014). The MMD has strong applications to two-sample testing, where it serves as the statistic for testing the hypothesis that  $X$  and  $Y$  are sampled from the same distribution (Gretton et al. 2012); this has applications in, for example, comparing microarray data from different experimental situations or in matching attributes when merging databases.

The MMK estimate can clearly be approximated with an explicit embedding: if  $k(x, y) \approx z(x)^\top z(y)$ ,

$$\begin{aligned}
\text{MMK}_z(X, Y) &= \frac{1}{nm} \sum_{i=1}^n \sum_{j=1}^m z(X_i)^\top z(Y_j) \\
&= \left( \frac{1}{n} \sum_{i=1}^n z(X_i) \right)^\top \left( \frac{1}{m} \sum_{j=1}^m z(Y_j) \right) \\
&= \bar{z}(X)^\top \bar{z}(Y).
\end{aligned}$$

Thus the biased estimator of MMK( $X, X$ ) is just  $\|\bar{z}(X)\|^2$ ; the unbiased estimator is

$$\frac{n^2}{n^2 - n} \left( \|\bar{z}(X)\|^2 - \frac{1}{n^2} \sum_{i=1}^n \|z(X_i)\|^2 \right)$$

When  $z(x)^\top z(x) = 1$ , as with  $\tilde{z}$ , this simplifies to  $\frac{n}{n-1} \|\bar{z}(X)\|^2 - \frac{1}{n-1}$ . When that is not necessarily true, as with  $\check{z}$ , that simplification holds only in expectation.

This has been noticed a few times in the literature, e.g. by Li and Tsang (2011). Gretton et al. (2012) gives different linear-time test statistics based on subsampling the sum over pairs; this version avoids reducing the amount of data used in favor of approximating the kernel. Additionally, when using the MMK in a kernel method this approximation allows the use of linear solvers, whereas the other linear approximations must still perform some pairwise computation. Zhao and Meng (2014) compare the empirical performance of an approximation equivalent to  $\check{z}$  against other linear-time approximations for two-sample testing. They find it is slower than the MMD-linear approximation but far more accurate, while being more accurate and comparable in speed to a block-based  $B$ -test (Zaremba et al. 2013).

Zhao and Meng (2014) also state a simple uniform error bound on the quality of this approximation. Specifically, since we can write  $|\text{MMK}_z(X, Y) - \text{MMK}(X, Y)|$  as the mean of  $|f(X_i, Y_j)|$ , uniform error bounds on  $f$  apply directly to  $\text{MMK}_z$ , including to the unbiased version of  $\text{MMK}_z(X, X)$ . Moreover, since  $\text{MMD}^2(X, Y) = \text{MMK}(X, X) + \text{MMK}(Y, Y) - 2\text{MMK}(X, Y)$ , its error is at most 4 times  $\|f\|_\infty$ . The advantage of this bound is that it applies uniformly to all sample sets on the input space  $\mathcal{X}$ , which is useful when we use  $\text{MMK}$  for a kernel method.

For a single two-sample test, however, we can get a tighter bound. Consider  $X$  and  $Y$  fixed for now. Note that  $\mathbb{E}\text{MMK}_z(X, Y) = \text{MMK}(X, Y)$ , by linearity of expectation. The variance of  $\text{MMK}_z(X, Y)$  is exactly

$$\frac{1}{n^2 m^2} \sum_{i,j} \sum_{i',j'} \text{Cov}(s(X_i, Y_j), s(X_{i'}, Y_{j'})), \quad (12)$$

which can be evaluated using the formulas of Section 2.1 and so, viewed only as a function of  $D$ , is  $O(1/D)$ . Alternatively, we can use a bounded difference approach: viewing  $\text{MMK}_z(X, Y)$  as a function of the  $\omega_i$ s, changing  $\omega_i$  to  $\hat{\omega}_i$  changes the  $\text{MMK}$  estimate by

$$\left| \frac{1}{nm} \sum_{i=1}^n \sum_{j=1}^m \frac{2}{D} (\cos(\hat{\omega}_i^\top (X_i - Y_j)) - \cos(\omega_i^\top (X_i - Y_j))) \right|,$$

which is at most  $4/D$ . The bound for  $\check{z}$  is in fact the same here. Thus McDiarmid’s inequality tells us that for fixed sets  $X$  and  $Y$  and either  $z$ ,

$$\Pr(|\text{MMK}_z(X, Y) - \text{MMK}(X, Y)| \leq 2 \exp(-\frac{1}{8} D \varepsilon^2)).$$

Thus  $\mathbb{E}|\text{MMK}_z(X, Y) - \text{MMK}(X, Y)| \leq 2\sqrt{2\pi/D}$ . Similarly,  $\text{MMD}_z$  can be changed by at most  $16/D$ , giving

$$\Pr(|\text{MMD}_z(X, Y) - \text{MMD}(X, Y)| \leq 2 \exp(-\frac{1}{128} D \varepsilon^2))$$

and expected absolute error of at most  $8\sqrt{2\pi/D}$ .

Now, if we consider the distributions  $P$  and  $Q$  to be fixed but the sample sets random, Theorems 7 and 10 of Gretton et al. (2012) give exponential convergence bounds for the biased and unbiased population estimators of  $\text{MMD}$ , which can easily be combined with the above bounds. Note that this approach allows the domain  $\mathcal{X}$  to be unbounded, unlike the other bound. One could extend this to a bound uniform over some smoothness class of distributions using the techniques of Section 2.2, though we do not do so here.

## 4 NUMERICAL EVALUATION

### 4.1 APPROXIMATION ON AN INTERVAL

We first conduct a detailed study of the approximations on the interval  $\mathcal{X} = [-b, b]$ . Specifically, we

evenly spaced 1000 points on  $[-5, 5]$  and approximated the kernel matrix using both embeddings at  $D \in \{50, 100, 200, \dots, 900, 1\,000, 2\,000, \dots, 9\,000, 10\,000\}$ , repeating each trial 1000 times, estimating  $\|f\|_\infty$  and  $\|f\|_\mu$  at those points. We do not consider  $d > 1$  here, because obtaining a reliable estimate of  $\sup|f|$  becomes very computationally expensive even for  $d = 2$ .

Figure 3 shows the behavior of  $\mathbb{E}\|f\|_\infty$  as  $b$  increases for various values of  $D$ . As expected, the  $\check{z}$  embeddings have almost no error near 0. The error increases out to one or two bandwidths, after which the curve appears approximately linear in  $\ell/\sigma$ , as predicted by Proposition 3.

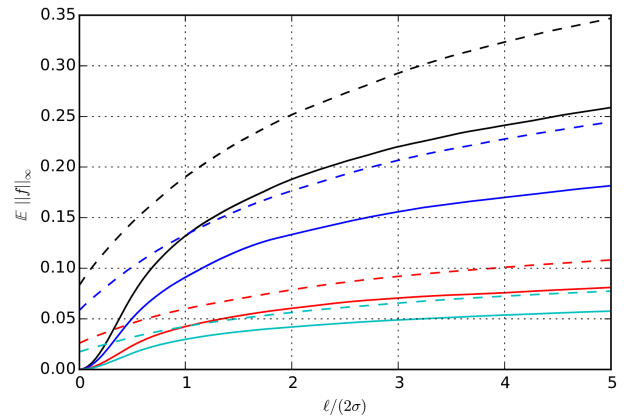


Figure 3: The maximum error within a given radius in  $\mathbb{R}$ , averaged over 1000 evaluations. Solid lines represent  $\check{z}$  and dashed lines  $\check{z}$ ; black is  $D = 50$ , blue is  $D = 100$ , red  $D = 500$ , and cyan  $D = 1000$ .

Figure 4 fixes  $b = 3$  and shows the expected maximal error as a function of  $D$ . It also plots the expected error obtained by numerically integrating the bounds of Propositions 1 and 2 (using the minimum of 1 and the bound). We can see that all of the bounds are fairly loose, but that the first version of the bound in the propositions (with  $\beta_d$ , the exponent depending on  $d$ , and  $\alpha_\varepsilon$ ) is substantially tighter than the second version when  $d = 1$ .

The bounds on  $\mathbb{E}\|f\|_\infty$  of Propositions 3 and 4 are unfortunately too loose to show on the same plot. However, one important property does hold. For a fixed  $\mathcal{X}$ , (8) predicts that  $\mathbb{E}\|f\|_\infty = O(1/\sqrt{D})$ . This holds empirically: performing linear regression of  $\log \mathbb{E}\|\tilde{f}\|_\infty$  against  $\log D$  yields a model of  $\mathbb{E}\|\tilde{f}\|_\infty = e^c D^m$ , with a 95% confidence interval for  $m$  of  $[-0.502, -0.496]$ ;  $\|\check{f}\|_\infty$  gives  $[-0.503, -0.497]$ . The integrated bounds of Propositions 1 and 2 do not fit the scaling as a function of  $D$  nearly as well.

Figure 5 shows the empirical survival function of the max error for  $D = 500$ , along with the bounds of Propositions 1 and 2 and those of Propositions 5 and 6 using the empirical mean. The latter bounds are tighter than the former for low  $\varepsilon$ , especially for low  $D$ , but have a lower slope.

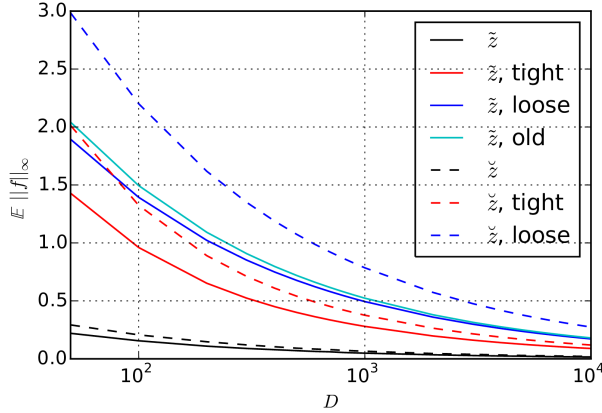


Figure 4:  $\mathbb{E}\|f\|_\infty$  for the Gaussian kernel on  $[-3, 3]$  with  $\sigma = 1$ , based on the mean of 1000 evaluations and on numerical integration of the bounds from Propositions 1 and 2. (“Tight” refers to the bound with constants depending on  $d$ , and “loose” the second version; “old” is the version from Rahimi and Recht (2007).)

The mean of the mean squared error, on the other hand, exactly follows the expectation of Section 2.3 using  $\mu$  as the uniform distribution on  $\mathcal{X}^2$ : in this case,  $\mathbb{E}\|f\|_\mu \approx 0.66/D$ ,  $\mathbb{E}\|f\|_\mu \approx 0.83/D$ . (This is natural, as the expectation is exact.) Convergence to that mean, however, is substantially faster than guaranteed by the McDiarmid bound of Propositions 7 and 8. We omit the plot due to space constraints.

## 4.2 MAXIMUM MEAN DISCREPANCY

We now turn to the problem of computing the MMD with a Fourier embedding. Specifically, we consider the problem of distinguishing the standard normal distribution  $\mathcal{N}(0, I_p)$  from the two-dimensional mixture  $0.95\mathcal{N}(0, I_2) + 0.05\mathcal{N}(0, \frac{1}{4}I_2)$ . We take fixed sample sets  $X$  and  $Y$  each of size 1000 and compute the biased MMD estimate with varying  $D$  for both  $\tilde{z}$  and  $\check{z}$ , we used a Gaussian kernel of bandwidth 1. The mean absolute errors of the resulting estimates are shown in Figure 6.  $\tilde{z}$  performs mildly better than  $\check{z}$ .

Again, the McDiarmid bound of Section 3.3 predicts that the mean absolute error decays as  $O(1/\sqrt{D})$ , but with too high a multiplicative constant; the 95% confidence interval for the exponent of  $D$  is  $[-0.515, -0.468]$  for  $\tilde{z}$  and  $[-0.520, -0.486]$  for  $\check{z}$ . We also know that the expected root mean squared error decays like  $O(1/\sqrt{D})$  via (12).

## 5 DISCUSSION

We provide a novel investigation of the approximation error of the popular random Fourier features, tightening ex-

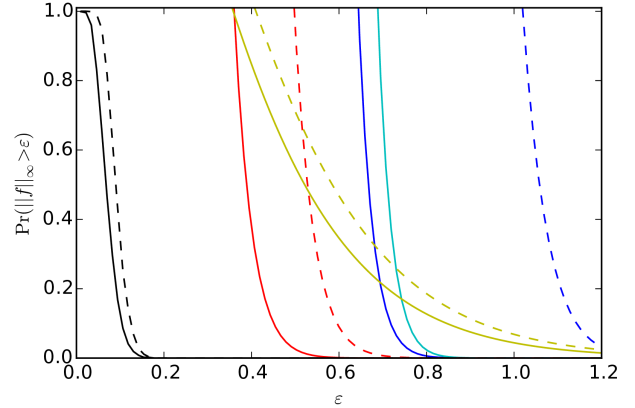


Figure 5:  $\Pr(\mathbb{E}\|f\|_\infty > \varepsilon)$  for the Gaussian kernel on  $[-3, 3]$  with  $\sigma = 1$  and  $D = 500$ , based on 1000 evaluations (black), numerical integration of the bounds from Propositions 1 and 2 (same colors as Figure 4), and the bounds of Propositions 5 and 6 using the empirical mean (yellow).

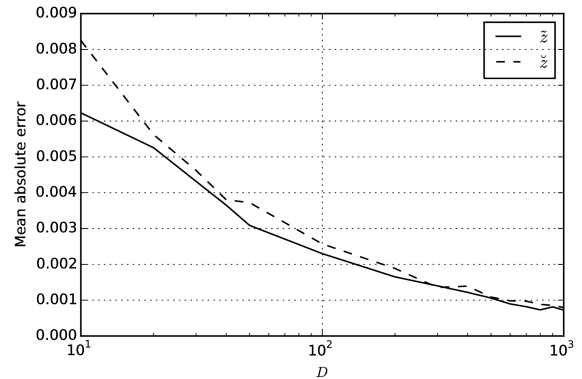


Figure 6: Mean absolute error of the biased estimator for  $\text{MMD}(X, Y)$ , based on 100 evaluations.

isting bounds and showing new ones, including an analytic bound on  $\mathbb{E}\|f\|_\infty$  and exponential concentration about its mean, as well as an exact form for  $\mathbb{E}\|f\|_\mu$  and exponential concentration in that case as well. We also extend previous results on the change in learned models due to kernel approximation. We verify some aspects of these bounds empirically for the Gaussian kernel. We also point out that, of the two embeddings provided by Rahimi and Recht (2007), the  $\tilde{z}$  embedding (with half as many sampled frequencies, but no additional noise due to phase shifts) is superior in the most common case of the Gaussian kernel.

## Acknowledgments

This work was funded in part by DARPA grant FA87501220324. DJS is also supported by a Sandia Campus Executive Program fellowship.



## References

- Bernstein, Sergei (1924). “On a modification of Chebyshev’s inequality and of the error formula of Laplace”. Russian. In: *Ann. Sci. Inst. Savantes Ukraine, Sect. Math.* 1, pp. 38–49.
- Bochner, Salomon (1959). *Lectures on Fourier integrals*. Princeton University Press.
- Boucheron, Stéphane, Gábor Lugosi, and Pascal Massart (2013). *Concentration Inequalities: A Nonasymptotic Theory of Independence*. Oxford, UK: Oxford University Press.
- Bousquet, Olivier (2002). “A Bennett concentration inequality and its application to suprema of empirical processes”. In: *Comptes Rendus Mathématique* 334, pp. 495–500.
- Bousquet, Olivier and André Elisseeff (2001). “Algorithmic Stability and Generalization Performance”. In: *Advances in Neural Information Processing Systems*, pp. 196–202.
- Cheng, Steve (2013). *Differentiation under the integral sign*. Version 16. URL: <http://planetmath.org/differentiationundertheintegralsign>.
- Cortes, Corinna, M Mohri, and A Talwalkar (2010). “On the impact of kernel approximation on learning accuracy”. In: *International Conference on Artificial Intelligence and Statistics*, pp. 113–120.
- Cucker, Felipe and Steve Smale (2001). “On the mathematical foundations of learning”. In: *Bulletin of the American Mathematical Society* 39.1, pp. 1–49.
- Dai, Bo et al. (2014). “Scalable Kernel Methods via Doubly Stochastic Gradients”. In: *Advances in Neural Information Processing Systems*, pp. 3041–3049.
- Dudley, Richard M (1967). “The sizes of compact subsets of Hilbert space and continuity of Gaussian processes”. In: *Journal of Functional Analysis* 1.3, pp. 290–330.
- Gretton, Arthur, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alex J Smola (2012). “A Kernel Two-Sample Test”. In: *The Journal of Machine Learning Research* 13.
- Hoeffding, Wassily (1963). “Probability inequalities for sums of bounded random variables”. In: *Journal of the American Statistical Association* 58.301, pp. 13–30.
- Joachims, Thorsten (2006). “Training linear SVMs in linear time”. In: *ACM SIGKDD international conference on Knowledge Discovery and Data mining*.
- Li, Shukai and Ivor W Tsang (2011). “Learning to Locate Relative Outliers”. In: *Asian Conference on Machine Learning*. Vol. 20. JMLR: Workshop and Conference Proceedings, pp. 47–62.
- McDiarmid, Colin (1989). “On the method of bounded differences”. In: *Surveys in combinatorics* 141.1, pp. 148–188.
- Muandet, Krikamol, Kenji Fukumizu, Francesco Dinuzzo, and Bernhard Schölkopf (2012). “Learning from distributions via support measure machines”. In: *Advances in Neural Information Processing Systems*.
- Pedregosa, F. et al. (2011). “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12, pp. 2825–2830.
- Raff, Edward (2011-15). *JSAT: Java Statistical Analysis Tool*. <https://code.google.com/p/java-statistical-analysis-tool/>.
- Rahimi, Ali and Benjamin Recht (2007). “Random Features for Large-Scale Kernel Machines”. In: *Advances in Neural Information Processing Systems*. MIT Press.
- (2008a). “Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning”. In: *Advances in Neural Information Processing Systems*. MIT Press, pp. 1313–1320.
- (2008b). “Uniform approximation of functions with random bases”. In: *46th Annual Allerton Conference on Communication, Control, and Computing*, pp. 555–561.
- Saunders, C., A. Gammerman, and V. Vovk (1998). “Ridge Regression Learning Algorithm in Dual Variables”. In: *Proceedings of the 15th International Conference on Machine Learning*, pp. 515–521.
- Sonnenburg, Sören et al. (2010). “The SHOGUN Machine Learning Toolbox”. In: *Journal of Machine Learning Research* 11, pp. 1799–1802.
- Yang, Tianbao, Yu-Feng Li, Mehrdad Mahdavi, Rong Jin, and Zhi-Hua Zhou (2012). “Nyström Method vs Random Fourier Features: A Theoretical and Empirical Comparison”. In: *Advances in Neural Information Processing Systems*. MIT Press.
- Yoshikawa, Yuya, Tomoharu Iwata, and Hiroshi Sawada (2014). “Latent Support Measure Machines for Bag-of-Words Data Classification”. In: *Advances in Neural Information Processing Systems*, pp. 1961–1969.
- Zaremba, Wojciech, Arthur Gretton, and Matthew Blaschko (2013). “B-tests: Low Variance Kernel Two-Sample Tests”. In: *Advances in Neural Information Processing Systems*.
- Zhao, Ji and Deyu Meng (2014). “FastMMD: Ensemble of Circular Discrepancy for Efficient Two-Sample Test”. In: arXiv: [1405.2664](https://arxiv.org/abs/1405.2664).

---

# Bayesian Structure Learning for Stationary Time Series

---

**Alex Tank**

University of Washington  
alextank@uw.edu

**Nicholas J. Foti**

University of Washington  
nfoti@uw.edu

**Emily B. Fox**

University of Washington  
ebfox@uw.edu

## Abstract

While much work has explored probabilistic graphical models for independent data, less attention has been paid to time series. The goal in this setting is to determine conditional independence relations between entire time series, which for stationary series, are encoded by zeros in the inverse spectral density matrix. We take a Bayesian approach to structure learning, placing priors on (i) the graph structure and (ii) spectral matrices given the graph. We leverage a Whittle likelihood approximation and define a conjugate prior—the *hyper complex inverse Wishart*—on the complex-valued and graph-constrained spectral matrices. Due to conjugacy, we can analytically marginalize the spectral matrices and obtain a closed-form marginal likelihood of the time series given a graph. Importantly, our analytic marginal likelihood allows us to avoid inference of the complex spectral matrices themselves and places us back into the framework of standard (Bayesian) structure learning. In particular, combining this marginal likelihood with our graph prior leads to efficient inference of the time series graph itself, which we base on a stochastic search procedure, though any standard approach can be straightforwardly modified to our time series case. We demonstrate our methods on analyzing stock data and neuroimaging data of brain activity during various auditory tasks.

## 1 INTRODUCTION

Probabilistic graphical models (PGMs)—which compactly encode a set of conditional independence statements—have become a defacto tool for defining probabilistic models over large sets of random variables. When faced with time series, dynamic Bayesian networks (DBNs) are commonly deployed and specify sparse between- and within-time de-

pendencies, often encoded by a *template model* replicated across time to straightforwardly model the growing set of random variables [1]. Learning template models requires specifying the set of dependency lags to be considered [2, 3]. In many applications, one instead aims to infer conditional independence between entire data streams accounting for interactions at all possible lags, represented by a *time series graphical model* (TGM). For example, imagine recording brain activity from multiple regions of the brain over time. Inference of a TGM in this setting would provide insight into the functional connectivity of different brain regions, an object of substantial scientific interest [4, 5]. TGMs have also been applied to intensive care monitoring [6] and financial time series [7].

The pioneering work of Dahlhaus [8] introduced the concept of undirected graphical models for stationary time series. The key insight was to transform the series to the *frequency domain* and express the graph relationships in the resulting spectral representation. For jointly Gaussian stationary time series, Dahlhaus [8] showed that conditional independencies between time series are encoded by zeros in the inverse spectral density matrices. This result is the frequency-domain analog to Gaussian graphical modeling in the i.i.d. (non-time-series) setting, where zeros in the inverse covariance matrix, or *precision matrix*, encode the conditional independencies between observed dimensions [9]. Dahlhaus’ insight was first exploited to perform independent hypothesis tests of conditional independence between each pair of time series [8], with more recent work correcting for multiple comparisons [10, 11].

A likelihood-based approach leveraging the *Whittle approximation* [12] has also been introduced [13]. The Whittle approximation casts the likelihood in the frequency domain with terms depending on the spectral density matrices critical to TGM structure learning, and independently so across frequencies. One approach scores graphs using AIC [13]. A recent penalized likelihood variant [14] places a joint graphical lasso [15] across frequencies to enforce a common zero pattern in the spectral density matrices. A penalized likelihood approach restricted to finite vector au-

to regressive processes has also been considered [7].

We instead consider a Bayesian approach to TGM structure learning, with all the benefits garnered from the Bayesian paradigm, including modeling within a generative framework where information from multiple sources can be integrated and combined with available prior knowledge. For example, neural data are notoriously noisy, and robust inferences often rely on integrating time series across multiple trials and individuals or recording platforms (e.g., EEG/MEG). Our approach also leverages the Whittle likelihood. We then introduce a novel hyper Markov law [16], the *hyper complex inverse Wishart* distribution, that serves as a conjugate prior for the spectral density matrices whose inverses have a zero pattern specified by a graph. For decomposable graphs, this formulation leads to a closed-form expression for the marginal likelihood of a multivariate time series given a graph. By placing a prior on graph structures, we achieve a fully Bayesian approach to TGM structure learning for stationary time series. For our graph prior, we consider a multiplicity correcting prior [17]. Our analytic expression for the marginal likelihood is critical to the practicality of our approach since we can avoid inference of the large set of high-dimensional, complex spectral density matrices. In particular, for a length  $T$  series of dimension  $p$ , there are  $T$   $p \times p$  spectral matrices to consider. In the i.i.d. setting, inference of just a single  $p \times p$  graph-constrained covariance matrix is challenging; in this setting, inference of the  $T$   $p \times p$  matrices is prohibitive.

Hyper Markov laws based on the hyper inverse Wishart are a popular tool for Bayesian graphical model selection in the i.i.d. setting [18, 19]. Indeed, many powerful Bayesian structure learning algorithms based on this framework have been developed, both for decomposable [20, 21] and non-decomposable [22, 23] graphs. By framing TGM structure learning in this common framework, we are able to apply existing state-of-the-art inference machinery for standard structure learning to the time series case. In this paper we use the feature-inclusion stochastic search (FINCS) procedure [20] for inference in decomposable models; however, many other MCMC and search schemes may be used. Importantly, future computational advances in Bayesian inference for i.i.d. graphical models may be easily extended using our framework to the time series case.

We test our methods on data simulated from vector autoregressive models with randomly generated TGMs. Our approach reaches almost perfect TGM recovery as the length of the time series or number of independent replicates increases. We then demonstrate the utility of our methods on a global stock indices dataset and MEG neuroimaging data of auditory attention switching tasks. In both cases we find meaningful, intuitive structure in the data.

Our paper is organized as follows. We provide background on graphical models and stationary time series in Sec. 2.

Our proposed TGM method is in Sec. 3, first introduced in the context of multiple independent realizations and then adapted to perform efficient inference of the TGM from only a single realization. In Sec. 5, we discuss how existing Bayesian structure learning methods may be modified to fit our formulation. Simulated results are in Sec. 6, with our stock and MEG analyses in Secs. 7 and 8, respectively.

## 2 BACKGROUND

### 2.1 Graphs

Let  $G = (V, E)$  be an undirected graph with vertex set  $V = \{1, \dots, p\}$  and edge set  $E$ , where  $E \subset \{(i, j) \in V \times V : i \neq j\}$ . Nodes  $i$  and  $j$  are adjacent, or *neighbors*, if  $(i, j) \in E$ . A *complete graph* is one having  $(i, j) \in E$  for every  $i, j \in V$  and complete subgraphs  $C \subset V$  are termed *cliques*. A triple of subgraphs  $(A, S, B)$  where  $V = A \cup B$  and  $S = A \cap B$  with  $S$  complete is called a *decomposition* if every path from a node in  $A$  to a node in  $B$  must pass through  $S$ , the *separator*. Recursively decomposing  $A$  and  $B$  in this fashion results in the *prime components* of a graph. If the prime components are complete then the graph is *decomposable*. We let the sets  $\mathcal{C} = \{C_1, \dots, C_K\}$  and  $\mathcal{S} = \{S_2, \dots, S_K\}$  each denote the prime components and their separators, respectively, generated by the decomposition. For simplicity, we restrict our attention to decomposable graphs but stress that our formulation is extensible to the non-decomposable case (see Sec. 9).

### 2.2 Hyper Markov distributions

For a given set of random variables  $X$ , with realization  $x \in \mathcal{X}$ , dimensionality  $p$ , and joint density  $p(x)$ , an undirected graphical model  $G$  can be constructed by stating that an edge  $(i, j) \notin E$  if  $X_i$  and  $X_j$  are conditionally independent given the remaining variables, i.e.  $X_j \perp\!\!\!\perp X_i | X_{Z_{ij}}$  where  $Z_{ij} = V \setminus \{i, j\}$ . If the graph is decomposable, the joint density decomposes over cliques and separators:

$$p(x) = \frac{\prod_{C \in \mathcal{C}} p(x_C)}{\prod_{S \in \mathcal{S}} p(x_S)} \quad (1)$$

where  $p(x_A)$  for  $A \subset V$  denotes the marginal distribution of the set of variables  $x_A$ .

A hyper Markov law [16] is a distribution over probability measures that is concentrated on distributions that obey the Markov properties specified by  $G$ . Examples include the hyper Wishart and hyper Dirichlet distribution [16, 18]. Such distributions have proven pivotal in Bayesian graphical modeling by serving as conjugate priors for the graph parameters conditioned on the graph structure  $G$ . For example, in Gaussian graphical models (GGMs), the hyper inverse Wishart distribution provides a conjugate prior for covariance matrices that obey a zero pattern in the precision, as specified by  $G$ . By integrating over the hyper

Markov distribution, one can obtain the *marginal likelihood* of the data conditioned on the structure  $G$  alone.

### 2.3 Stationary time series

Let  $X(t) = (X_1(t), \dots, X_p(t))^T \in \mathbb{R}^p$  for  $t \in \mathbb{Z}$  be a multivariate Gaussian stationary time series such that:

$$E(X(t)) = \mu \quad \forall t \in \mathbb{Z} \quad (2)$$

$$\text{Cov}(X(t), X(t+h)) = \Gamma(h) \quad \forall t, h \in \mathbb{Z}. \quad (3)$$

A time series probabilistic graphical model (TGM),  $G = (V, E)$ , may be constructed by letting  $(i, j) \notin E$  denote that the entire time series  $X_i(\cdot)$  and  $X_j(\cdot)$  are conditionally independent given the remaining collection of time series  $X_{Z_{ij}}$  where  $Z_{ij} = V \setminus \{i, j\}$ . For the Gaussian stationary series we consider, one can show that conditional independence holds between time series iff [8]

$$\text{Cov}(X_i(t), X_j(t+h) | X_{Z_{ij}}) = 0 \quad \forall h \in \mathbb{Z}. \quad (4)$$

The *spectral density matrix* of a stationary time series is defined as the Fourier transform of the lagged covariance matrices,  $\Gamma(h) = \text{Cov}(X(t), X(t+h))$ :

$$S(\lambda) = \sum_{h=-\infty}^{\infty} \Gamma(h) e^{-i\lambda h} \quad (5)$$

for  $\lambda \in [0, 2\pi]$  and  $S(\lambda) \in \mathbb{C}^{p \times p}$  and Hermitian positive definite. The marginal dependencies between time series are captured by  $S(\lambda)$ , and from Eq. (5),  $S(\lambda)_{ij} = 0$  for all  $\lambda \in [0, 2\pi]$  iff  $\Gamma(h)_{ij} = 0$  for all  $h \in \mathbb{Z}$ . Furthermore, conditional independence between Gaussian stationary time series holds iff

$$S(\lambda)_{ij}^{-1} = 0 \quad \forall \lambda \in [0, 2\pi], \quad (6)$$

implying that inferring zeros in the inverse spectral density matrices across frequencies equates with inferring the TGM structure [8]. More background on the spectral approach to time series is presented in the Supplement.

## 3 A BAYESIAN APPROACH

There are two standard approaches to Bayesian inference in graphical models: (1) placing a prior that jointly specifies the graph structure and associated parameters or (2) placing a prior on graph structures and then a prior on parameters given a graph; both rely on specifying a likelihood model. We opt for the second approach and describe the various components in this section. At a high level, our methods combine existing Whittle likelihood based methods [13, 14] with the hyper Markov framework to Bayesian graphical modeling [19, 18]. In the context of our TGMs, we introduce a conjugate *hyper complex inverse Wishart* prior on graph-constrained spectral density matrices. By integrating out the spectral density matrices, we obtain a

marginal likelihood of the time series given the graph structure,  $G$ , allowing us to straightforwardly leverage state-of-the-art computational methods for i.i.d. Bayesian structure learning.

### 3.1 Whittle likelihood

Let  $\mathbf{X} = [X(1), \dots, X(T)]$ , with  $x(t) \in \mathbb{R}^p$  a realization of a  $p$ -dimensional stationary Gaussian time series observed at  $T$  time points, and  $\mathbf{X}_{1:N} = \{\mathbf{X}^1, \dots, \mathbf{X}^N\}$  be the collection of  $N$  independent realizations. We move to the frequency domain by transforming each  $\mathbf{X}^i$  using a discrete Fourier transform. Let  $d_{nk} \in \mathbb{C}^p$  denote the discrete Fourier coefficient associated with the  $n$ th time series at frequency  $\lambda_k = \frac{2\pi k}{T}$ :

$$d_{nk} = \frac{1}{T} \sum_{t=0}^{T-1} x_n(t) e^{-i\lambda_k t}. \quad (7)$$

The Whittle approximation [12] assumes the Fourier coefficients are independent *complex normal random variables* with mean zero and covariance given by the corresponding spectral density matrix  $S_k = S(\lambda_k)$ :

$$d_{nk} \sim \mathcal{N}_c(0, S_k) \quad k = 0, \dots, T-1, \quad (8)$$

such that the likelihood of  $\mathbf{X}_{1:N}$  is approximated as

$$p(\mathbf{X}_{1:N} | S_{0:T-1}) \approx \prod_{n=1}^N \prod_{k=0}^{T-1} \frac{1}{\pi^p |S_k|} e^{-d_{nk}^* S_k^{-1} d_{nk}}, \quad (9)$$

where  $\frac{1}{\pi^p |S|} e^{-z^* S^{-1} z}$  is the density of a complex normal distribution,  $\mathcal{N}_c(0, S)$ , with  $S \in \mathbb{C}^{p \times p}$  and Hermitian positive definite. See the Supplement. The Whittle approximation holds asymptotically with large  $T$  [24, 25, 12]. This approximation has been used in the Bayesian context in [26, 27]

Recall that conditional independencies are encoded in the off diagonal elements of  $S_k^{-1}$ . If time series  $X_i(t)$  and  $X_j(t)$  are conditionally independent, then the Whittle approximation says that as  $T$  gets large the  $i$ th and  $j$ th elements of the Fourier coefficients  $d_{nk}$  are conditional independent across all frequencies. Thus, if  $G$  is decomposable, Eq. (9) can be rewritten as

$$p(\mathbf{X}_{1:N} | G, S_{0:(T-1)}) \approx \prod_{k=0}^{T-1} \frac{\prod_{C \in \mathcal{C}} \frac{1}{\pi^{N|C|} |S_{kC}|^N} e^{-\text{tr} P_{kC} S_{kC}^{-1}}}{\prod_{S \in \mathcal{S}} \frac{1}{\pi^{N|S|} |S_{kS}|^N} e^{-\text{tr} P_{kS} S_{kS}^{-1}}} \quad (10)$$

where

$$P_k = \sum_{n=1}^N d_{nk} d_{nk}^* \quad (11)$$

is the aggregate *periodogram* over the  $N$  time series at frequency  $\frac{2\pi k}{T}$ . For  $A \subset V$ ,  $S_{kA}$  and  $P_{kA}$  are the restriction of both matrices to the elements in  $A$  and  $|A|$  denotes the cardinality of the set  $A$ .

### 3.2 Hyper complex inverse Wishart prior on graph-constrained spectral density matrices

We seek a prior for the spectral density matrices,  $S_k$ , whose inverses each have zeros dictated by a graph  $G$ . Recall that these  $S_k$  matrices are complex-valued and restricted to be Hermitian positive definite. As discussed in Sec. 2.2, the hyper inverse Wishart distribution serves as a prior for real-valued, positive-definite matrices with pre-specified zeros in the inverse, and is a conjugate prior for the covariance of a zero-mean GGM. Motivated by the connection between GGMs and our TGMs, and the analogous structure of our TGM-based Whittle likelihood of Eq. (10) to that of a GGM with  $N$  i.i.d. observations, we propose a novel *hyper complex inverse Wishart* prior with density function

$$p(\Sigma|\delta, W, G) \propto \mathbf{1}_{\Sigma \in M^+(G)} |\Sigma|^{-(\delta+2p)} e^{-\text{tr}W\Sigma^{-1}} \quad (12)$$

for *degrees of freedom*  $\delta > 0$ , *scale matrix*  $W \in \mathbb{C}^{p \times p}$  positive definite and Hermitian, and graph  $G$ . We have used an analogous parameterization to that of the hyper inverse Wishart [16]. Here,  $\Sigma \in M^+(G)$  denotes that  $\Sigma$  is in the set of all Hermitian positive-definite matrices with  $(\Sigma^{-1})_{ij} = 0$  for all  $(i, j) \notin E$ . When  $G$  is decomposable, the normalization constant is available and the density decomposes over cliques and separators:

$$\begin{aligned} p(\Sigma|\delta, W, G) &= \frac{\prod_{C \in \mathcal{C}} \text{IW}_c(\Sigma_C|\delta, W_C)}{\prod_{S \in \mathcal{S}} \text{IW}_c(\Sigma_S|\delta, W_S)} \quad (13) \\ &= \frac{\prod_{C \in \mathcal{C}} B(W_C, \delta) |\Sigma_C|^{-(\delta+2|C|)} e^{-\text{tr}W_C\Sigma_C^{-1}}}{\prod_{S \in \mathcal{S}} B(W_S, \delta) |\Sigma_S|^{-(\delta+2|S|)} e^{-\text{tr}W_S\Sigma_S^{-1}}}, \quad (14) \end{aligned}$$

where  $\text{IW}_c$  denotes the complex inverse Wishart [25] detailed in the Supplement with normalizer

$$B(W, \delta) = \frac{|W|^{\delta+p}}{\pi^{\frac{p(p-1)}{2}} \prod_{j=1}^p (\delta+p-j)!}. \quad (15)$$

We denote our proposed prior as  $HIW_c(\delta, W, G)$  and specify

$$S_k | G \sim HIW_c(\delta_k, W_k, G) \quad k = 0, \dots, T-1. \quad (16)$$

In the Supplement, we show that this prior specification is *conjugate* to the TGM-based Whittle likelihood of Eq. (10). Also note that the graph,  $G$ , is shared across all frequencies.

### 3.3 Marginal likelihood

Due to conjugacy of our proposed hyper complex inverse Wishart prior, the marginal likelihood of the time series  $\mathbf{X}_{1:N}$  given a decomposable graph  $G$ , integrating out the spectral density matrices  $S_{0:T-1}$ , has a closed form which is derived in the Supplement and given by

$$p(\mathbf{X}_{1:N}|G) \approx \pi^{-NTp} \prod_{k=0}^{T-1} \frac{h(W_k, \delta_k, G)}{h(W_k^*, \delta_k^*, G)}. \quad (17)$$

Here,  $\delta_k^* = \delta_k + N$ ,  $W_k^* = W_k + P_k$ , and

$$h(W, \delta, G) = \frac{\prod_{C \in \mathcal{C}} B(W_C, \delta)}{\prod_{S \in \mathcal{S}} B(W_S, \delta)}. \quad (18)$$

From the definition of  $\delta_k^*$ , we see that  $N$ , the number of time series, acts as the effective number of observations in this case. For the i.i.d. GGM,  $N$  represents the number of independent vector-valued observations; in our TGM,  $N$  plays the same role, but represents the number of independent *time series* observations. Likewise, as in standard inverse Wishart based modeling of covariances for i.i.d. Gaussian data, based on a set of  $N$  i.i.d. complex normal observations of Fourier coefficients  $d_{nk}$  with covariance  $S_k$  (see Eq. (9)), we update the prior scale matrix  $W_k$  with the outer product  $P_k = \sum_{n=1}^N d_{nk}d_{nk}^*$ , which is the aggregate *periodogram* (see Eq. (11)).

Having an analytic marginal likelihood of the time series given a PGM allows us to perform inference directly over graphs, sidestepping any thorny issues with inference directly on the  $T p \times p$  spectral density matrices themselves. This is a critical feature of the practicality of our approach.

### 3.4 Fractional priors for model selection

Marginal likelihoods used for model comparison [28] are notoriously sensitive to the choice of prior parameters, or *hyperparameters*. In our case, the marginal likelihood in Eq. (17) depends strongly on the hyper complex inverse Wishart scale matrix,  $W_k$ . Since the scale and shape of the spectral density matrices are not known a priori, and vary dramatically across frequencies, we employ *fractional priors* [29] over each  $S_k$ . Fractional priors effectively hold out some fraction of the data, and utilize that fraction to determine an adequate hyperparameter setting for each model. The rest of the data are then used for model comparison. Fractional priors have been deployed for graphical model selection in i.i.d. graphs and have a number of desirable properties such as information consistency and demonstrated robustness [20]. In our case, under a fractional prior with parameter  $g \in (0, 1)$ , the fractional marginal likelihood is

$$p(\mathbf{X}_{1:N}|g, G) = \pi^{-NTp} \prod_{k=0}^{T-1} \frac{h(gP_k, gN, G)}{h(P_k, N, G)}. \quad (19)$$

Here, we see that  $g$  controls the fraction of data used for prior formulation versus model comparison. Importantly, we now have just a single, scalar, and interpretable parameter  $g$  to tune. Default settings are suggested in [29, 20].

### 3.5 Graph prior

There are two common approaches in the literature to specifying a prior distribution on graphs. The first approach places a uniform distribution on the space of all possible

graphs [18, 30, 31]. As noted in [32], this prior puts high weight on graphs with a medium number of edges and significantly less weight on graphs with small or many edges. In response to this problem, it has been proposed to place a prior directly on the size of the graph and then consider a conditionally uniform prior on all graphs of the same size [32, 33, 19]. We follow this later approach and place a binomial distribution on the number of edges,  $k$ :

$$p(G) \propto r^k (1-r)^{m-k}, \quad (20)$$

where  $r$  is the prior probability that each of  $m = \frac{p(p-1)}{2}$  possible undirected edges  $(i, j) \in V \times V$  is included. Since  $r$  is unknown, we further place a  $\text{Beta}(a, b)$  prior over  $r$ . Integrating out  $r$  gives the marginal prior over graphs

$$p(G) \propto \frac{\beta(a+k, b+m-k)}{\beta(a, b)} \quad (21)$$

where  $\beta(\cdot, \cdot)$  is the beta function. As explored in [20], this is a multiplicity correcting prior [34] over graphs with the desirable property of diminishing false positive edge discoveries as extra unconnected nodes are added to the graph.

## 4 METHODS FOR SINGLE TIME SERIES

In some applications of interest one observes only a single multivariate time series,  $N = 1$ , from which the graph must be inferred. Two challenges arise in this setting: (1) the effective number of observations informing Eq. (17) is just one and (2) the periodogram used in computing  $W_k^*$  is noisy regardless of the length of the series,  $T$ . The periodogram is a notoriously poor estimator of the spectral density, and when the spectral density itself is of primary interest, a common frequentist method is to smooth the periodogram to obtain a consistent spectral density estimator [14, 13, 8]. One could imagine using the smoothed periodogram as a plug-in estimator in Eq. (17), scaled by the effective degrees of freedom (see the Supplement for more details on this plug-in estimator for our formulation). An alternative variance-reduction technique is the Bartlett method [35], that divides the length  $T$  series into  $M$  shorter series of length  $\frac{T}{M}$  and averages the resulting  $M$  periodograms, but at the cost of reduced resolution (i.e., number of considered frequencies). This approach mimics the implicit smoothing that occurs when we compute the periodogram based on  $N$  truly independent series each of length  $T$ , as in Eq. (11).

In contrast to a plug-in estimator, a natural Bayesian approach enforces smoothing across frequencies via a prior distribution over the set of spectral densities [26]. Previous approaches have coupled elements of a Cholesky decomposition of each spectral density matrix across frequencies, however this approach is unsuitable to our case since

1) it does not enforce sparsity in the inverse spectral density and 2) a prior of this form will remove the simple marginal likelihood structure in Eq. (17) that we harness for efficient inference. Motivated by our aims to both share information across frequencies and maintain the form of the marginal, we utilize a piecewise constant prior over spectral densities given a graph,  $G$ . We partition the interval  $[0, 2\pi]$  into  $M$  intervals  $w_1 = [0, \frac{2\pi}{M})$ ,  $\dots$ ,  $w_j = [\frac{2\pi(j-1)}{M}, \frac{2\pi j}{M})$ ,  $\dots$ ,  $w_M = [\frac{2\pi(M-1)}{M}, 2\pi]$  and then draw a separate positive definite Hermitian matrix from a  $HIW_c$  distribution for each interval:

$$\tilde{S}_j \sim HIW_c(\delta, W_j, G) \quad j = 1, \dots, M. \quad (22)$$

Our resulting spectral density is simply

$$S(\lambda) = \sum_{j=1}^M \mathbf{1}_{\lambda \in w_j} \tilde{S}_j \quad \forall \lambda \in [0, 2\pi]. \quad (23)$$

Under this prior, the marginal likelihood for the single ( $N = 1$ ) time series becomes

$$p(\mathbf{X}|G) \approx \pi^{-Mp} \prod_{j=1}^M \frac{h(W_j, \delta_j, G)}{h(W_j^*, \delta_j^*, G)} \quad (24)$$

where  $\delta_j^* = \delta_j + \sum_{k=0}^{T-1} \mathbf{1}_{\lambda_k \in w_j}$  and  $W_j^* = W_j + \sum_{k=0}^{T-1} \mathbf{1}_{\lambda_k \in w_j} P_k$ . By setting  $M = \lfloor \sqrt{T} \rfloor$ , we obtain an asymptotically approximate nonparametric prior distribution over continuous spectral density matrices: for  $T$  large enough the prior puts positive support on spectral density matrices arbitrarily close to any continuous spectral density over  $[0, 2\pi]$ . Furthermore, under this setting as  $T \rightarrow \infty$ , the number of Fourier frequencies, and thus number of samples  $\sum_{k=0}^{T-1} \mathbf{1}_{\lambda_k \in w_j}$ , within each interval grows as  $\sqrt{T}$ .

## 5 INFERENCE

Bayesian structural learning algorithms for decomposable graphs come in two flavors: MCMC samplers and stochastic search procedures [20, 22]. By placing decomposable graphical inference for time series in the same framework as for the i.i.d. case via our analytic  $p(\mathbf{X}_{1:N} | G)$ , we can easily modify both types of existing methods for the time series case.

Classical MCMC samplers for decomposable graphs sample from the posterior over graphs via Metropolis-Hastings (MH) by proposing single edge addition and deletion moves that keep the graph decomposable [18, 32]. While it is possible to obtain any decomposable graph from any other decomposable graph via a sequence of edge additions and deletions, the path may be hard to reach leading to prohibitive converge times for even a moderate number of vertices  $p$ . More recent graph samplers add more global moves by either randomly generating new decomposable graphs [36] or by generating from a Markov chain

over a junction tree representation of the graph [21]. To compute the MH acceptance ratio, these samplers rely on computing ratios of present and proposed marginal likelihoods. For simple edge additions and deletions, this ratio simplifies into a function of only the cliques and separators that change between moves. For our case, the ratio expands into a product over frequencies of the same affected cliques and separators, allowing simple modifications to the existing implementations of these samplers to handle TGMs.

All current MCMC samplers struggle to scale to even moderate numbers of nodes. In contexts where point estimates suffice, we can instead consider stochastic search procedures. We utilize a modification of the efficient feature-inclusion stochastic search (FINCS) [20] for inference in our TGMs. FINCS interleaves three moves: 1) single edge addition and deletion moves for local changes to the graph, 2) global sampling moves where edges are added independently to an empty graph and the final graph is triangulated to maintain decomposability, and 3) resampling at step  $t$  a full graph from a list of past visited models,  $\{G_1, G_2, \dots, G_{t-1}\}$ , in proportion to their posterior probabilities. In steps 1) and 2), to enforce exploration of high probability regions, edge additions that tend to continually improve the model probability are preferentially selected in proportion to a current heuristic estimate of the posterior edge probability

$$\hat{q}_{ij}(t) = \frac{\sum_{k=1}^t 1_{\{i,j\} \in E_t} p(X_{1:N}|G_t) p(G_t)}{\sum_{k=1}^t p(X_{1:N}|G_t)}, \quad (25)$$

where  $E_t$  is the current edge set. Edge deletions are performed proportional to  $\hat{q}_{ij}^{-1}(t)$ . As in MCMC samplers [18, 32], the junction tree representation of the graph can be efficiently updated after each local move since the two graphs only differ by a single clique and its corresponding separators, allowing a quick computation of the marginal likelihood of a proposed graph in Eq. (17). Importantly, the FINCS algorithm depends on the data only through the marginal likelihoods of the cliques  $C$ —used to compute the full graph marginal likelihood—which in our TGM case is a product over  $T$  frequencies:

$$\prod_{k=0}^{T-1} \frac{B(W_{k,C}, \delta)}{B(W_{k,C}^*, \delta^*)}. \quad (26)$$

That is, our implementation simply modifies the original FINCS definition of the clique marginal likelihood.

## 6 SIMULATIONS

To test our TGM methods, we consider simulated setups for both  $N > 1$  and  $N = 1$  time series each generated from an order-1 vector autoregressive process, denoted VAR(1), for  $p = 20$  dimensions. Specifically, we simulated data from the model

$$x(t) = Ax(t-1) + \epsilon(t), \quad (27)$$

where  $x(t) \in \mathbb{R}^p$ ,  $A \in \mathbb{R}^{p \times p}$ , and  $\epsilon(t) \sim N(0, I_{p \times p})$ . The inverse spectral density of a VAR(1) process is given by [7]

$$S(\lambda)^{-1} = I + A^T A + e^{-i\lambda} A + e^{i\lambda} A^T. \quad (28)$$

Random sparse TGMs were generated by first restricting  $A$  to be upper triangular. Following the simulated setup in [7], we set the diagonal elements to a constant  $A_{ii} = .5$  and sample the upper diagonal elements as  $a_{ij} \sim .5\delta_{ij}$ , where  $\delta_{ij} \sim \text{Binomial}(\rho)$  with  $\rho = .2$  for all simulations. The graph  $G$  was then determined by identifying the zeros in  $S(\lambda)^{-1}$  using Eq. (28). Proposed  $A$  matrices were accepted when both the absolute value of all eigenvalues of  $A$  were less than one, making the series stationary, and the graph  $G$  determined by  $A$  was decomposable.

We note that since our formulation reduces to a standard structure learning problem, our emphasis is less on assessing performance with respect to  $p$ , which should follow from whichever structure learning algorithm is selected; instead, our focus is on  $N$  and  $T$ , which are specific to the time series and spectral analysis. For example, in the FINCS algorithm [20], it is quoted that the method can handle graphs with up to roughly  $p = 100$  nodes.

### 6.1 Multiple time series

To analyze how our TGM structure learning performance varies with the number of time series replicates,  $N$ , we simulated data for  $N \in \{20, 50, 100, 150, 200, 250, 300, 350\}$  and  $T \in \{25, 50, 100, 500, 1000, 1500, 2000\}$ . This process was repeated 200 times for each combination of  $N$  and  $T$ . Each time series is first decomposed into its discrete Fourier components. We then ran 10,000 iterations of the FINCS algorithm using the fractional marginal likelihood in Eq. (19) with  $g = \frac{4}{N}$ , a default setting [29, 20]. Our graph prior followed the multiplicity correcting form in Eq. (21) with  $a = b = 1$ . The graph visited with highest posterior probability was then selected and true and false positive rates were computed. Results are displayed in Fig. 1. Across  $T$ , the true positive rate increases quickly with the number of series,  $N$ , achieving an almost perfect true positive rate by about  $N = 150$ . We also see that the rate of increase in the true positive rate increases with the length of the series  $T$ , which relates to the number of considered Fourier frequencies. It is interesting to note that for all  $T$  under consideration, the false positive rate tends to start very low ( $\approx .005$ ) for  $N = 20$  replicates then spike at  $N \in \{50, 100\}$  before declining again. This occurs due to the fact that at low  $N$ , very few edges are introduced at all, perhaps due to an Occam’s razor type effect of marginal likelihoods penalizing model complexity. As  $N$  starts to increase, more edges are introduced, both correct and incorrect, and as  $N$  further increases, the false edges are pruned and true edges are retained, leading to a decline in the false positive rate. Note that the false positive spike tends to be more pronounced for time series of

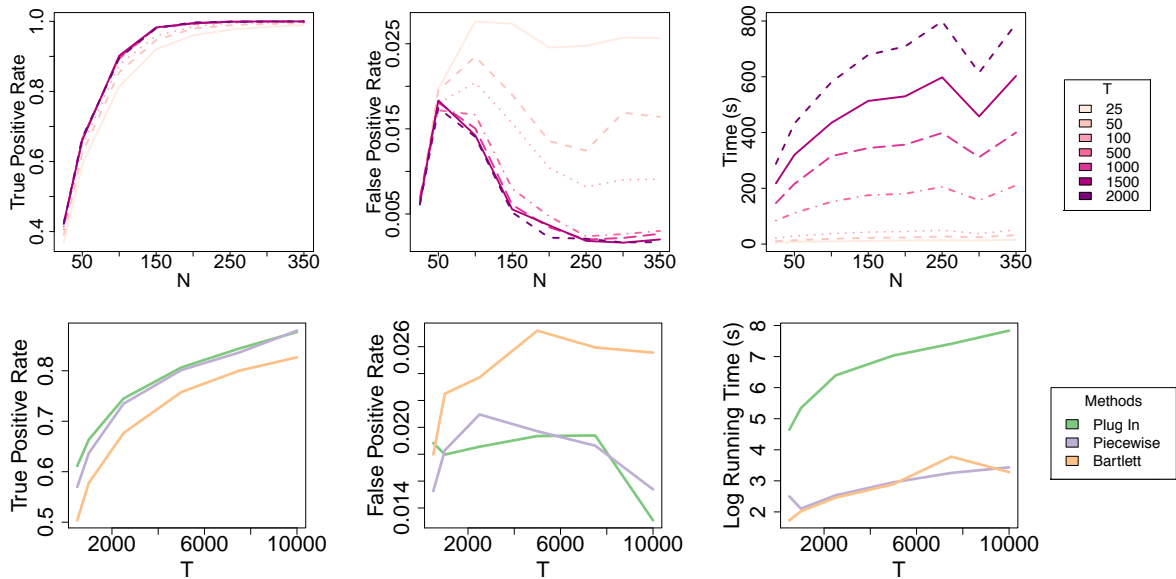


Figure 1: **Top:** As a function of the number of time series  $N$ , and plotted for various values of their length  $T$ , (left) mean true positive rate, (middle) median false positive rate, and (right) mean running time computed across the 200 replicates. Standard error bars are small relative to the scale of the plots and are omitted for clarity. **Bottom:** Same plots as a function of  $T$  for a single time series ( $N = 1$ ), and plotted for various periodogram smoothing techniques.

smaller length,  $T \in \{25, 50\}$ . One would expect to see significant improvements, especially for small  $N$ , by leveraging the piecewise constant prior of Sec. 4 and explored in Sec. 6.2 where we show that we are able to learn graphs from just  $N = 1$  time series. However, we chose not to include this prior in this analysis so as not to confound its effect with our performance. Here, the noisy periodogram is smoothed implicitly by averaging over  $N$ .

Finally, in Fig. 1 we see that runtime increases as a function of  $T$  due to the dependence on  $T$  in the marginal likelihood computation of Eq. (17), though significant cost reductions can be achieved through parallelizations leveraging the product form.

## 6.2 Single time series: comparison of methods

To assess the performance of our single-time-series methods outlined in Sec. 4, we simulated a time series with  $T \in \{500, 1000, 2500, 5000, 7500, 10000\}$ . For the piecewise constant prior method, we use  $M = \lfloor \sqrt{T} \rfloor$  pieces. We compare against the Bartlett time-series-splitting approach with the number of splits set to  $\lfloor \sqrt{T} \rfloor$ . We also examine a smoothed plug-in estimator of the spectral density using a Daniell smoother outlined in the Supplement with  $m = \lfloor \frac{\sqrt{T}}{2} \rfloor$  for a total window size of  $2 \lfloor \frac{\sqrt{T}}{2} \rfloor + 1 \approx \lfloor \sqrt{T} \rfloor$ . For each method, the FINCS algorithm was run for 10,000 iterations and the highest scoring graph was selected and used to compute true and false positive rates. This process was repeated 200 times with results displayed in Fig. 1

with a replicate representative of our median performance shown in Fig. 2. The true positive rate increases for all three methods as a function of  $T$ , achieving a final value of about .9 for both the plug-in and piecewise constant prior methods and .79 for the Bartlett method at  $T = 10000$ . All methods maintain a low false positive rate around .02. Overall, the Bartlett method performs uniformly worse in terms of both true and false positive performance, while the piecewise prior method performs on par with the plug-in method, but at a fraction of the computational cost. Further experimental simulations are given in the Supplement.

## 7 GLOBAL STOCK INDICES

We explore the utility of our method in discovering conditional independencies between countries inherent in the global financial system. A similar experiment was conducted in [7] using a penalized-likelihood approach to learn TGMs, but restricted to finite-order VAR models with pre-specified order. (Recall that our method only assumes Gaussian stationarity, which includes the class of possibly infinite order VAR processes.) Using [www.globalfinancialdata.com](http://www.globalfinancialdata.com), we acquired the daily closing prices of 17 stock indices in US dollars for various countries around the world (see the Supplement for the full list) from June 3, 1997 to June 30, 1999. Missing prices were backfilled and only days where all exchanges traded were considered which resulted in time series of length 542. Following standard practice when analyzing stock prices, we converted the closing prices,  $p_t$ , on day  $t$  to log-returns according to



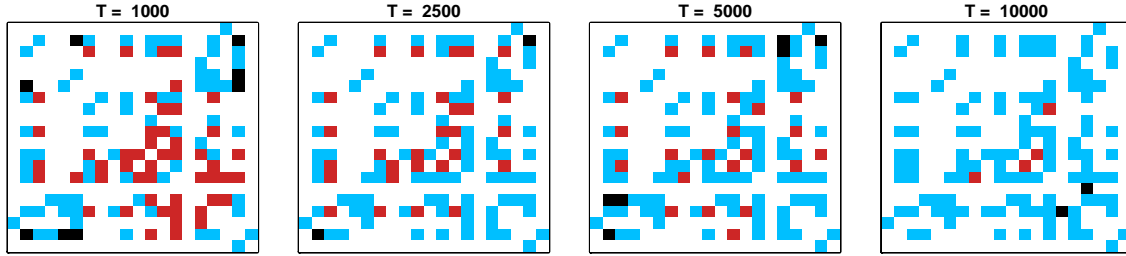


Figure 2: Example evolution of error types for the piecewise prior method as a function of series length,  $T \in \{1000, 2500, 5000, 10000\}$  and  $N = 1$ , for a selected graph. **Blue, red, black,** and white entries indicate true positives, false negatives, false positives, and true negatives, respectively. The graph was selected by choosing the graph out of 200 replications with median true positive rate at  $T = 2500$ .

$$r_t = 100 \log(p_t/p_{t-1}).$$

We compare the graphical models inferred under two settings: (i) treating the log-returns as independent (as in [20]) and (ii) using our methods to learn a TGM treating the log-returns as a time series. The best graphical models learned in each scenario are depicted in Fig. 3.

For our TGM algorithm, we computed the periodogram for the 17-dimensional time series, resulting in 542 complex-valued matrices of dimension  $17 \times 17$ . Since we only have one realization of the time series, we smoothed the periodogram using the techniques and settings discussed in Sec. 6.2. We then ran the FINCS algorithm for 100,000 iterations. We compare the resulting highest-probability graph (see Fig. 3) to that learned treating the time series as independent based on the model in [20], again using 100,000 iterations of the FINCS algorithm, but in its originally proposed form for non-temporal data.

In Figure 3, we see that in both cases we recover some geographical relationships between countries. However, the independent model returns a significantly denser graph than that learned by our TGM approach. Since the independent model is not taking the temporal nature of the data into account, some edges are likely spurious due to random correlations. The TGM, on the other hand, provides an interpretable and intuitive structure with strong geographic connections. For example, there is a distinct United Kingdom / eurozone cluster of Germany ‘DE’, Finland ‘FI’, Netherlands ‘NL’, Belgium ‘BE’, Switzerland ‘CH’, Austria ‘AT’, Spain ‘ES’, Italy ‘IT’, Portugal ‘PT’, and the United Kingdom ‘UK’. Another distinct cluster includes the United States ‘US’, Canada ‘CA’, Hong Kong ‘HK’ (whose currency is linked to the USD), and Australia ‘AU’ (whose currency is correlated with the US S&P), with Japan ‘JP’ hanging off this cluster. One perhaps strange missing link is between Ireland ‘IE’ and the UK, though the US and Ireland have a long history of economic connections possibly explaining why Ireland is included in the separator between these two distinct clusters.

In the Supplement, we include (i) a comparison of our

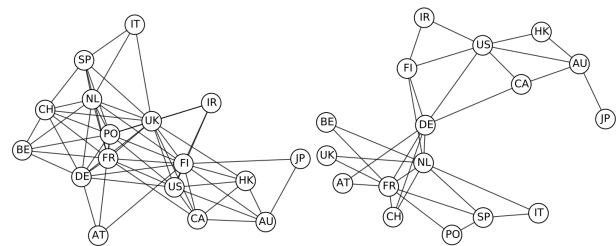


Figure 3: Graphical models with the highest posterior probability for the stock index data. **Left:** Treating the log-returns as independent. **Right:** Using our TGM algorithm. In both cases, we see regional connections, but our TGM algorithm results in a sparser and more interpretable graph.

learned graph with that of Songsiri et. al. [7], and (ii) further details on the stock data itself.

## 8 MAGNETOENCEPHALOGRAPHY DATA

Next we learn TGMs to capture the structure of underlying cortical dynamics from magnetoencephalography (MEG) data collected from ten subjects who were asked to perform a task while maintaining focus on an audio stream and then again while switching focus [37]. Our goal is to discover differences in the underlying TGMs between the non-switching and switching attention conditions. Such differences provide further understanding into the neural underpinnings of auditory selective selection, an important constituent to communication.

The data were collected for each subject performing the experiment in the *switching* (S) and *non-switching* (N) attention conditions. For both S and N conditions, each subject performed the task under an auditory condition of *high* (U) and *low* (D) pitch, and spatial conditions of *left* (L) and *right* (R) attending. For each of the eight possible conditions, MEG recordings were collected resulting in a 150-dimensional time series of length 992 where each dimen-

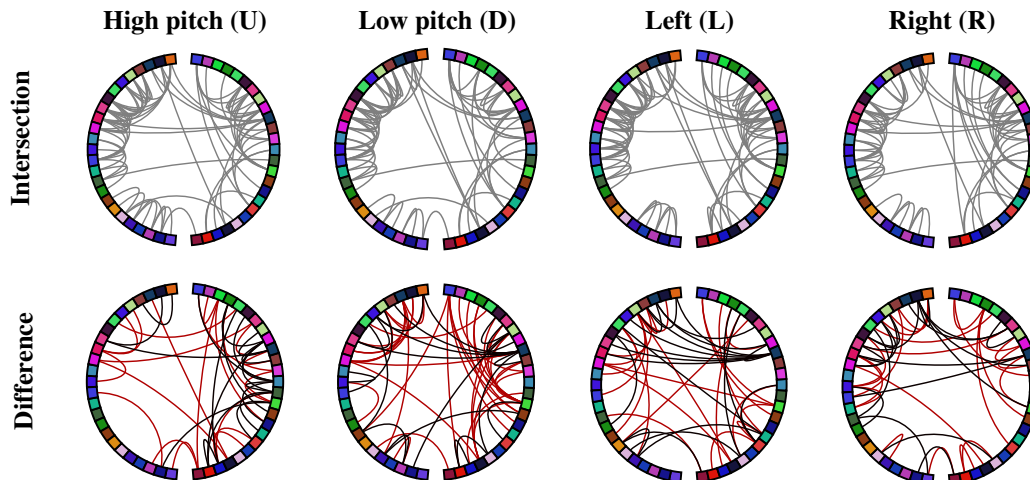


Figure 4: Learned TGMs for different MEG conditions. Each node on the periphery represents a brain region with location indicating anatomical location. **Top:** Intersection of learned edges between switching and non-switching conditions. **Bottom:** *Black* edges indicating those in the non-switching condition but not in the switching and *red* vice versa.

sion corresponds to a localized region of the brain. We have between 17 and 30 trials for each subject, resulting in about 200 replicate time series per condition.

Often with MEG data, many of the dimensions are dominated by noise due to limited brain activity in that region. We reduced the number of brain regions we studied from 150 to 50 by only considering those with largest variance. In particular, for each trial we mean-centered all of the time-series and computed the variance and retained the top 50 most volatile regions.

We computed the periodogram for each trial and averaged across trials within each condition, resulting in eight periodograms. We ran our spectral TGS version of the FINCS algorithm on these periodograms for 100,000 iterations with fractional prior parameter  $4/N_c$ , where  $N_c$  is the number of trials for condition  $c \in \{S, N\} \times \{U, D, L, R\}$ . We also ran the algorithm for 1.7 million iterations and saw no difference in the resulting graphs.

In Figure 4, we depict the intersections and differences between the learned graphs for each experimental condition. We see in the top row that there are a lot of shared connections between the switching and non-switching conditions for each auditory condition. In the bottom row, the differences between the switching and non-switching conditions are depicted where red edges are those in the switching condition but not the non-switching, and black edges are the reverse. The difference plots show that there seems to be substantial “rewiring” for many of the conditions with many edges connecting frontal to back regions. Interestingly, we again see consistencies in these rewirings across conditions. Additionally, we reliably uncover local connections between adjacent brain regions across experimental conditions. Such observations provide guidance for developing experiments and methods to discern the underlying mechanisms that give rise to these different structures.

## 9 DISCUSSION

We introduced a Bayesian approach to graphical model structure learning for time series. In particular, we propose a prior—the *hyper complex inverse Wishart* distribution—for the spectral density matrices in a Whittle likelihood approximation. For decomposable graphs, this prior is conjugate and leads to a closed-form expression of the marginal likelihood of the time series given the graph, marginalizing the spectral density matrices across frequencies. Being able to integrate out this large collection of complex matrices—one for each time point—is critical to developing a practical and scalable inference algorithm. For this, exploiting the fact that our marginal likelihood is analogous to that for i.i.d. Gaussian graphical models [19] but with a product over the number of Fourier frequencies, allows us to deploy straightforward modifications to existing MCMC and stochastic search algorithms. Our simulations show that when many time series are observed, our method recovers the correct graph. When a single time series is observed, we proposed a method to increase robustness of our graph estimation using a piecewise constant prior. Our results on the stock and MEG datasets demonstrated our ability to discover intuitive and interpretable structure in these datasets, importantly leveraging the temporal dependencies.

Extensions to non-decomposable graphs are possible using the i.i.d. graph approaches in both [31] and [22]. A Laplace approximation to the marginal likelihood for non-decomposable graphs is proposed in [22], which we could similarly utilize to approximate the frequency-specific marginal at each term in Equation (17). Parallelizing the Laplace approximation computation across frequencies would lead to a scalable method for inference in non-decomposable time series graphs.

**Acknowledgements:** This work was supported by DARPA Grant FA9550-12-1-0406 negotiated by AFOSR, ONR Grant N00014-10-1-0746, NSF CAREER Award IIS-1350133, and AFOSR Grant FA9550-12-1-0453.

## References

- [1] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.
- [2] B. D. Ziebart, A. K. Dey, and J. A. Bagnell. Learning selectively conditioned forest structures with applications to dbns and classification. *UAI*, 2007.
- [3] M. R. Siracusa and J. W. Fisher III. Tractable Bayesian inference of time-series dependence structure. In *AISTATS*, 2009.
- [4] O. Sporns. *Networks of the Brain*. MIT Press, 2010.
- [5] T. Medkour, A. T. Walden, Burgess A. P., and Strelets V. B. Brain connectivity in positive and negative syndrome schizophrenia. *Neuroscience*, 169(4):1779 – 1788, 2010.
- [6] U. Gather, M. Imhoff, and R. Fried. Graphical models for multivariate time series from intensive care monitoring. *Statistics in Medicine*, 21(18), 2002.
- [7] J. Songsiri and L. Vandenberghe. Topology selection in graphical models of autoregressive processes. *JMLR*, 11:2671–2705, 2010.
- [8] R. Dahlhaus. Graphical interaction models for multivariate time series. *Metrika*, 51(2):157–172, 2000.
- [9] S. L. Lauritzen. *Graphical Models*. Oxford University Press, 1996.
- [10] Y. Matsuda. A test statistic for graphical modelling of multivariate time series. *Biometrika*, 93(2):pp. 399–409, 2006.
- [11] R. J. Wolstenholme and A. T. Walden. An efficient approach to graphical modeling of time series. *ArXiv e-prints*, 2015.
- [12] P. Whittle. The analysis of multiple stationary time series. *JRSS(B)*, 15(1):125–139, 1953.
- [13] F. R. Bach and M. I. Jordan. Learning graphical models for stationary time series. *IEEE Transactions on Signal Processing*, 52(8):2189–2199, 2004.
- [14] A. Jung, G. Hannak, and N. Görtz. Graphical LASSO based model selection for time series. *ArXiv e-prints*, 2014.
- [15] P. Danaher, P. Wang, and D. M. Witten. The joint graphical lasso for inverse covariance estimation across multiple classes. *JRSS(B)*, 76(2):373–397, 2014.
- [16] A. P. Dawid and S. L. Lauritzen. Hyper Markov laws in the statistical analysis of decomposable graphical models. *Ann. Statist.*, 21(3):1272–1317, 1993.
- [17] C. M. Carvalho and J. G. Scott. Objective Bayesian model selection in Gaussian graphical models. *Biometrika*, 96(3):497–512, 2009.
- [18] P. Giudici and P. J. Green. Decomposable graphical Gaussian model determination. *Biometrika*, 86(4):785–801, 1999.
- [19] B. Jones, C. Carvalho, A. Dobra, C. Hans, C. Carter, and M. West. Experiments in stochastic computation for high-dimensional graphical models. *Statistical Science*, 20(4):388–400, 2005.
- [20] J. G. Scott and C. M. Carvalho. Feature-inclusion stochastic search for Gaussian graphical models. *J. Comp. Graph. Stat.*, 17(4):790–808, 2008.
- [21] P. J. Green and A. Thomas. Sampling decomposable graphs using a Markov chain on junction trees. *Biometrika*, 100(1):91–110, 2013.
- [22] M. Baback, E. Khan, K. M. Murphy, and B. M. Marlin. Accelerating Bayesian structural inference for non-decomposable Gaussian graphical models. In *NIPS*, pages 1285–1293, 2009.
- [23] A. Mohammadi and E. C. Wit. Bayesian structure learning in sparse Gaussian graphical models. *Bayesian Analysis*, 10(1):109–138, 2015.
- [24] P. J. Brockwell and R. A. Davis. *Time Series: Theory and Methods*. Springer-Verlag, New York, NY, 1991.
- [25] D.R. Brillinger. *Time Series: Data Analysis an Theory*. Holden-Day, 2001.
- [26] O. Rosen and D. S. Stoffer. Automatic estimation of multivariate spectra via smoothing splines. *Biometrika*, 94(2):335–345, 2007.
- [27] R. T. Krafty, O. Rosen, D. S. Stoffer, D. J. Buysse, and M. H. Hall. Conditional spectral analysis of replicated multiple time series with application to nocturnal physiology. *ArXiv e-prints*, 2015.
- [28] R. E. Kass and A. E. Raftery. Bayes factors. *Journal of the American Statistical Association*, 90:773–795, 1995.
- [29] A. O’Hagan. Fractional Bayes factors for model comparison. *JRSS(B)*, 57(1):99–138, 1995.
- [30] P. Dellaportas, P. Giudici, and G. Roberts. Bayesian inference for nondecomposable graphical Gaussian models. *Sankhy: The Indian Journal of Statistics*, 65(1):43–55, 2003.
- [31] A. Roverato. Hyper inverse Wishart distribution for non-decomposable graphs and its application to Bayesian inference for Gaussian graphical models. *Scand. J. Stat*, 29(3):391–411, 2002.
- [32] H. Armstrong, C. Carter, K. Wong, and R. Kohn. Bayesian covariance matrix estimation using a mixture of decomposable graphical models. *Statistics and Computing*, 19(3):303–316, 2009.
- [33] A. Dobra, C. Hans, B. Jones, J.R. Nevins, Joseph R., G. Yao, and M. West. Sparse graphical models for exploring gene expression data. *Journal of Multivariate Analysis*, 90(1):196–212, 2004.
- [34] J. G. Scott and J. O. Berger. An exploration of aspects of Bayesian multiple testing. *Journal of Statistical Planning and Inference*, 136:2144–2162, 2006.
- [35] M. S. Bartlett. Smoothing periodograms from time series with continuous spectra. *Nature*, 161(4096):686–687, 1948.
- [36] H. Zhu, N. Strawn, and D. B. Dunson. Bayesian graphical models for multivariate functional data. *ArXiv e-prints*, 2014.
- [37] E. Larson and A.K.C. Lee. Switching auditory attention using spatial and non-spatial features recruits different cortical networks. *NeuroImage*, 84:681–687, 2014.

---

# Learning from Pairwise Marginal Independencies

---

**Johannes Textor**

Theoretical Biology & Bioinformatics  
Utrecht University, The Netherlands  
johannes.textor@gmx.de

**Alexander Idelberger**

Theoretical Computer Science  
University of Lübeck, Germany  
alex@pirx.de

**Maciej Liśkiewicz**

Theoretical Computer Science  
University of Lübeck, Germany  
liskiewi@tcs.uni-luebeck.de

## Abstract

We consider graphs that represent pairwise marginal independencies amongst a set of variables (for instance, the zero entries of a covariance matrix for normal data). We characterize the directed acyclic graphs (DAGs) that faithfully explain a given set of independencies, and derive algorithms to efficiently enumerate such structures. Our results map out the space of faithful causal models for a given set of pairwise marginal independence relations. This allows us to show the extent to which causal inference is possible without using conditional independence tests.

## 1 INTRODUCTION

DAGs and other graphical models encode conditional independence (CI) relationships in probability distributions. Therefore, CI tests are a natural building block of algorithms that infer such models from data. For example, the PC algorithm for learning DAGs (Kalisch and Bühlmann, 2007) and the FCI (Spirtes et al., 2000) and RFCI (Colombo et al., 2012) algorithms for learning maximal ancestral graphs are all based on CI tests.

CI testing is still an ongoing research topic, to which the UAI community is contributing (e.g. Zhang et al., 2011; Doran et al., 2014). But at least for continuous variables, CI testing will always remain more difficult than testing marginal independence for quite fundamental reasons (Bergsma, 2004). Intuitively, the difficulty is that two variables  $x$  and  $y$  could be dependent “almost nowhere”, e.g., for only a few values of the conditioning variable  $z$ . This suggests a two-staged approach to structure learning: first try to learn as much as possible from simpler independence tests before applying CI tests. Here, we present a theoretical basis for extracting as much information as possible from the simplest kind of stochastic independence – pairwise marginal independence.

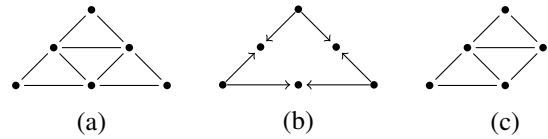


Figure 1: (a) A *marginal independence graph*  $\mathcal{U}$  whose missing edges represent pairwise marginal independencies. (b) A *faithful DAG*  $\mathcal{G}$  entailing the same set of pairwise marginal independencies as  $\mathcal{U}$ . (c) A graph for which no such faithful DAG exists.

More precisely, we will consider the following problem. We are given the set of pairwise marginal independencies that hold amongst some variables of interest. Such sets can be represented as graphs whose missing edges correspond to independencies (Figure 1a). We call such graphs *marginal independence graphs*. We wish to find DAGs on the same variables that entail exactly the given set of pairwise marginal independencies (Figure 1b). We call such DAGs *faithful*. Sometimes no such DAGs exist (e.g., Figure 1c). Else, we are interested in finding the set of *all* faithful DAGs, hoping that this set will be substantially smaller than the set of all possible DAGs on the same variables. Those candidate DAGs could then be probed further by using joint marginal or conditional independence tests.

Other authors have represented marginal (in)dependencies using bidirected graphs (Drton and Richardson, 2003; Richardson, 2003; Drton and Richardson, 2008b), instead of undirected graphs like we do here. We hope that the reader is compensated for this small departure from community standards by the lower amount of clutter in our figures, and the greater ease to link our work to standard graph theoretical results. We also emphasize that we model only pairwise, and not higher-order joint dependencies. However, for Gaussian data, pairwise independence entails joint independence. In that case, our marginal independence graphs are equivalent to *covariance graphs* (Cox and Wermuth, 1993; Pearl and Wermuth, 1994; Drton and Richardson, 2003, 2008a; Peña, 2013), whose missing edges represent zero covariances.

Our results generalize the work of Pearl and Wermuth (1994) who showed (but did not prove) how to find *some* faithful DAGs for a given covariance graph. We review these and other connections to related work in Section 3 where we also link our problem to the theory of partially ordered sets (posets). This connection allows us to identify certain maximal and minimal faithful DAGs. Based on these “boundary DAGs” we then derive a characterization of all faithful DAGs (Section 4), and construct related enumeration algorithms (Section 5). We use these algorithms to explore the combinatorial structure of faithful DAG models (Section 6) which leads, among other things, to a quantification of how much pairwise marginal independencies reduce structural causal uncertainty. Finally, we ask what happens when a set of independencies can *not* be explained by any DAG. How many additional variables will we need? We prove that this problem is NP-hard (Section 7).

Preliminary versions of many of the results presented in this paper were obtained in the Master’s thesis of the second author (Idelberger, 2014).

## 2 PRELIMINARIES

In this paper we use the abbreviation *iff* for the connective “if and only if”. A graph  $\mathcal{G} = (V, E)$  consists of a set of nodes (variables)  $V$  and set of edges  $E$ . We consider undirected graphs (which we simply refer to as graphs), directed graphs, and mixed graphs that can have both undirected edges (denotes as  $x - y$ ) and directed edges (denoted as  $x \rightarrow y$ ). Two nodes are *adjacent* if they are linked by any edge. A *clique* in a graph is a node set  $C \subseteq V$  such that all  $u, v \in C$  are adjacent. Conversely, an *independent set* is a node set  $I \subseteq V$  in which no two nodes  $u, v \in I$  are adjacent. A *maximal clique* is a clique for which no proper superset of nodes is also a clique. For any  $v \in V$ , the *neighborhood*  $N(v)$  is the set of nodes adjacent to  $v$  and the *boundary*  $Bd(v)$  is the neighborhood of  $v$  including  $v$ , i.e.  $Bd(v) = N(v) \cup \{v\}$ . A node  $v$  is called *simplicial* if  $Bd(v)$  is a clique. Equivalently,  $v$  is simplicial iff  $Bd(v) \subseteq Bd(w)$  for all  $w \in N(v)$  (Kloks et al., 2000). A clique that contains simplicial nodes is called a *simplex*. Every simplex is a maximal clique, and every simplicial node belongs to exactly one simplex. The *degree*  $d(v)$  of a node  $v$  is  $|N(v)|$ . If for two graphs  $\mathcal{G} = (V, E(\mathcal{G}))$  and  $\mathcal{G}' = (V, E(\mathcal{G}'))$  we have  $E(\mathcal{G}) \subseteq E(\mathcal{G}')$ , then  $\mathcal{G}$  is an *edge subgraph* of  $\mathcal{G}'$  and  $\mathcal{G}'$  is an *edge supergraph* of  $\mathcal{G}$ . The *skeleton* of a directed graph  $\mathcal{G}$  is obtained by replacing every edge  $u \rightarrow v$  by an undirected edge  $u - v$ .

A *path* of length  $n - 1$  is a sequence of  $n$  distinct nodes in which successive nodes are pairwise adjacent. A *directed path*  $x \rightarrow \dots \rightarrow y$  consists of directed edges that all point towards  $y$ . In a directed graph, a node  $u$  is an *ancestor* of another node  $v$  if  $u = v$  or if there is a directed path

$u \rightarrow \dots \rightarrow v$ . For each edge  $u \rightarrow v$ , we say that  $u$  is a *parent* of  $v$  and  $v$  is a *child* of  $u$ . If two nodes  $u, v$  in a directed graph have a common ancestor  $w$  (which can be  $u$  or  $v$ ), then the path  $u \leftarrow \dots \leftarrow w \rightarrow \dots \rightarrow v$  is called a *trek* connecting  $u$  and  $v$ . A DAG is called *transitive* if, for all  $u \neq v$ , it contains an edge  $u \rightarrow v$  whenever there is a directed path from  $u$  to  $v$ . Given a DAG  $\mathcal{G}$ , the *transitive closure* is the unique transitive graph that implies the same ancestor relationships as  $\mathcal{G}$ , whereas the *transitive reduction* is the unique edge-minimal graph that implies the same ancestor relationships.

In this paper we encounter several well-known graph classes, e.g., chordal graphs and trivially perfect graphs. We will give brief definitions when appropriate, but we direct the reader to the excellent survey by Brandstädt et al. (1999) for further details.

## 3 SIMPLE MARGINAL INDEPENDENCE GRAPHS

In this section we define the class of graphs which can be explained using a directed acyclic graph (DAG) on the same variables. We will refer to such graphs as *simple marginal independence graphs* (SMIGs).

**Definition 3.1.** A graph  $\mathcal{U} = (V, E(\mathcal{U}))$  is called the *simple marginal independence graph* (SMIG), or *marginal independence graph* of a DAG  $\mathcal{G} = (V, E(\mathcal{G}))$  if for all  $v, w \in V$ ,  $v - w \in E(\mathcal{U})$  iff  $v$  and  $w$  have a common ancestor in  $\mathcal{G}$ . If  $\mathcal{U}$  is the marginal independence graph of  $\mathcal{G}$  then we also say that  $\mathcal{G}$  is *faithful* to  $\mathcal{U}$ . **SMIG** is the set of all graphs  $\mathcal{U}$  for which there exists a faithful DAG  $\mathcal{G}$ . Note that each DAG has exactly one marginal independence graph.

Again, we point out that marginal independence graphs are often called (and drawn as) *bidirected graphs* in the literature, though the term “marginal independence graph” has also been used by various authors (e.g. Tan et al., 2014).

### 3.1 SMIGs and Dependency Models

In this subsection we recall briefly the general setting for modeling (in)dependencies proposed by Pearl and Verma (1987) and show the relationship between that model and SMIGs. In the definitions below  $V$  denotes a set of variables and  $X, Y$  and  $Z$  are three disjoint subsets of  $V$ .

**Definition 3.2** (Pearl and Verma (1987)). A *dependency model*  $\mathcal{M}$  over  $V$  is any subset of triplets  $(X, Z, Y)$  which represent independencies, that is,  $(X, Z, Y) \in \mathcal{M}$  asserts that  $X$  is independent of  $Y$  given  $Z$ .

A probabilistic dependency model  $\mathcal{M}_P$  is defined in terms of a probability distribution  $P$  over  $V$ . By definition  $(X, Z, Y) \in \mathcal{M}_P$  iff for any instantiation  $\hat{x}, \hat{y}$  and  $\hat{z}$  of the variables in these subsets  $P(\hat{x} \mid \hat{y} \hat{z}) = P(\hat{x} \mid \hat{z})$ .

A directed acyclic graph dependency model  $\mathcal{M}_{\mathcal{G}}$  is defined in terms of a DAG  $\mathcal{G}$ . By definition  $(X, Z, Y) \in \mathcal{M}_{\mathcal{G}}$  iff  $X$  and  $Y$  are  $d$ -separated by  $Z$  in  $\mathcal{G}$  (for a definition of  $d$ -separation by a set  $Z$  see Pearl and Verma (1987)).

We define a *marginal* dependency model, resp. marginal probabilistic and marginal DAG dependency model, analogously as Pearl and Verma (1987) with the restriction that the second component of any triple  $(X, Z, Y)$  is the empty set. Thus, such marginal dependency models are sets of pairs  $(X, Y)$ . It is easy to see that the following properties are satisfied.

**Lemma 3.3.** *Let  $\mathcal{M}$  be a marginal probabilistic dependency model or a marginal DAG dependency model. Then  $\mathcal{M}$  is closed under:*

*Symmetry:*  $(X, Y) \in \mathcal{M} \Leftrightarrow (Y, X) \in \mathcal{M}$  and

*Decomposition:*  $(X, Y \cup W) \in \mathcal{M} \Rightarrow (X, Y) \in \mathcal{M}$ .

*Moreover, if  $\mathcal{M}$  is a marginal DAG dependency model then it is also closed under*

*Union:*  $(X, Y), (X, W) \in \mathcal{M} \Rightarrow (X, Y \cup W) \in \mathcal{M}$ .

The marginal probabilistic dependency model is not closed under union in general. For instance, consider two independent, uniformly distributed binary variables  $y$  and  $w$  and let  $x = y \oplus w$ , where  $\oplus$  denotes xor of two bits. For the model  $\mathcal{M}_{\mathcal{P}}$  defined in terms of probability over  $x, y, w$  we have that  $(\{x\}, \{y\})$  and  $(\{x\}, \{w\})$  belong to  $\mathcal{M}_{\mathcal{P}}$  but  $(\{x\}, \{y, w\})$  does not.

In this paper we will *not* assume that the marginal independencies in the data are closed under union. Instead, we only consider pairwise independencies, which we formalize as follows.

**Definition 3.4.** *Let  $\mathcal{M}$  be a marginal probabilistic dependency model over  $V$ . Then the simple marginal independence graph  $\mathcal{U} = (V, E(\mathcal{U}))$  of  $\mathcal{M}$  is the graph in which  $x - y \in E(\mathcal{U})$  iff  $(\{x\}, \{y\}) \notin \mathcal{M}$ .*

Thus, in general, marginal independence graphs do not contain any information on higher-order *joint* independencies present in the data. However, under certain common parametric assumptions, dependency models would be closed under union as well. This holds, for instance, if the data are normally distributed. In that case, marginal independence is equivalent to zero covariance, pairwise independence implies joint independence, and marginal independence graphs become covariance graphs.

The following is not difficult to see.

**Proposition 3.5.** *A marginal dependency model  $\mathcal{M}$  which is closed under symmetry, decomposition, and union coincides with the transitive closure of  $\{(\{x\}, \{y\}) : x, y \in V\} \cap \mathcal{M}$  over symmetry and union.*

This Proposition entails that if the marginal dependencies in the data are closed under these properties, then the entire

marginal dependency model is represented by the marginal independence graph.

### 3.2 SMIGs and Partially Ordered Sets

To reach our aim of a complete and constructive characterization of the DAGs faithful to a given SMIG, it is useful to observe that marginal independence graphs are invariant with respect to the insertion or deletion of transitive edges from the DAG. We formalize this as follows.

**Definition 3.6.** *A (labelled) poset  $\mathcal{P}$  is a DAG that is identical to its transitive closure.*

**Proposition 3.7.** *The marginal independence graphs of a DAG  $\mathcal{G}$  and its transitive closure  $\mathcal{P}(\mathcal{G})$  are identical.*

*Proof.* Two nodes are not adjacent in the marginal independence graph iff they have no common ancestor in the DAG. Transitive edges do not influence ancestral relationships.  $\square$

We thus restrict our attention to finding *posets* that are faithful to a given SMIG. Note that faithful DAGs can then be obtained by deleting transitive edges from faithful posets; since no DAG obtained in this way can be an edge subgraph of two different posets, this construction is unique and well-defined. In particular, by deleting *all* transitive edges from a poset, we obtain a sparse graphical representation of the poset as defined below.

**Definition 3.8.** *Given a poset  $\mathcal{P} = (V, E)$ , its transitive reduction is the unique DAG  $\mathcal{G}_{\mathcal{P}} = (V, E')$  for which  $\mathcal{P}(\mathcal{G}) = \mathcal{P}$  and  $E'$  is the smallest set where  $E' \subseteq E$ .*

Transitive reductions are also known as *Hasse diagrams*, though Hasse diagrams are usually unlabeled. Different posets can have the same marginal independence graphs, e.g. the posets with Hasse diagrams  $\mathcal{P}_1 = x \rightarrow y \rightarrow z$  and  $\mathcal{P}_2 = x \leftarrow y \rightarrow z$ . Similarly, Markov equivalence is a sufficient but not necessary condition to inducing the same marginal independence graphs (adding an edge  $x \rightarrow z$  to  $\mathcal{P}_2$  changes the poset and the Markov equivalence class, but not the marginal independence graph).

### 3.3 Recognizing SMIGs

We first recall existing results that show which graphs admit a faithful DAG at all, and how to find such DAGs if possible. Note that many of these results have been stated without proof (Pearl and Wermuth, 1994), but our connection to posets will make some of these proofs straightforward. The following notion related to posets is required.

**Definition 3.9** (Bound graph (McMorris and Zaslavsky, 1982)). *For a poset  $\mathcal{P} = (V, E)$ , the bound graph  $\mathcal{B} = (V, E')$  of  $\mathcal{P}$  is the graph where  $x - y \in E'$  iff  $x$  and  $y$  share a lower bound, i.e., have a common ancestor in  $\mathcal{P}$ .*

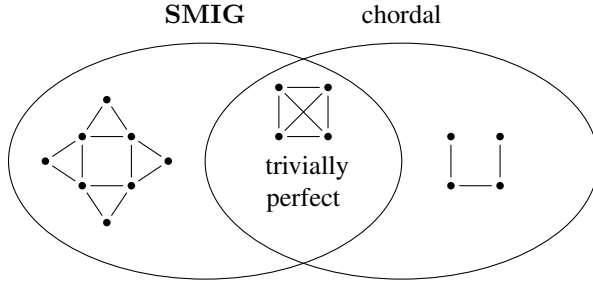


Figure 2: Relation between chordal graphs, trivially perfect graphs, and SMIG. In graph theory, SMIG is known as the class of (upper/lower) *bound graphs* (Cheston and Jap, 2006).

**Theorem 3.10.** SMIG is the set of all graphs for which every edge is contained in a simplex.

*Proof.* This is Theorem 2 in Pearl and Wermuth (1994) (who referred to simplexes as “exterior cliques”). Alternatively, we can observe that the marginal independence graph  $\mathcal{U}$  of a poset  $\mathcal{P}$  (Definition 3.1) is equal to its bound graph (Definition 3.9). The characterization of bound graphs as “edge simplicial” graphs has been proven by McMorris and Zaslavsky (1982) by noting that simplicial nodes in  $\mathcal{U}$  correspond to possible minimal elements in  $\mathcal{P}$ . We note that this result predates the equivalent statement in Pearl and Wermuth (1994).  $\square$

Though all bound graphs have a faithful poset, not all bound graphs have one with the same skeleton; see Figure 1a,b for a counterexample. However, the graphs for which a poset with the same skeleton can be found are nicely characterizable in terms of forbidden subgraphs.

**Theorem 3.11** (Pearl and Wermuth (1994)). *Given a graph  $\mathcal{U}$ , a DAG  $\mathcal{G}$  that is faithful to  $\mathcal{U}$  and has the same skeleton exists iff  $\mathcal{U}$  is trivially perfect (i.e.,  $\mathcal{U}$  has no  $P_4 = \text{---}$  nor a  $C_4 = \square$  as induced subgraph).*

It is known that the trivially perfect graphs are the intersection of the bound graphs and the chordal graphs (Figure 2; Cheston and Jap, 2006).

This nice result begs the question whether a similar characterization is also possible for SMIG. As the following observation shows, that is not the case.

**Proposition 3.12.** Every graph  $\mathcal{U}$  is an induced subgraph of some graph  $\mathcal{U}' \in \text{SMIG}$ .

*Proof.* Take any graph  $\mathcal{U} = (V, E)$  and construct a new graph  $\mathcal{U}'$  as follows. For every edge  $e = u - v$  in  $\mathcal{U}$ , add a new node  $v_e$  to  $V$  and add edges  $v_e - u$  and  $v_e - v$ . Obviously  $\mathcal{U}$  is an induced subgraph of  $\mathcal{U}'$ . To see that  $\mathcal{U}'$  is in SMIG, consider the DAG  $\mathcal{G}$  consisting of the nodes in  $\mathcal{U}'$  and the edges  $v \leftarrow v_e \rightarrow u$  and for each newly added

node in  $\mathcal{U}'$ . Then  $\mathcal{U}$  is the marginal independence graph of  $\mathcal{G}$ .  $\square$

The graph class characterization implies efficient recognition algorithms for SMIGs.

**Theorem 3.13.** It can be tested in polynomial time whether a graph  $\mathcal{U}$  is a SMIG.

*Proof.* Verifying the graphical condition of Theorem 3.10 amounts to testing whether all edges reside within a simplex. However, knowing that SMIGs are bound graphs, we can apply an efficient algorithm for bound graph recognition that uses radix sort and simplex elimination and achieves a runtime of  $\mathcal{O}(n + sm)$  (Skowrońska and Sysło, 1984), where  $s \leq n$  is the number of simplexes in the graph. This is typically better than  $\mathcal{O}(n^3)$  because large  $m$  implies small  $s$  and vice versa. Alternatively, we can apply known fast algorithms to find all simplicial nodes (Kloks et al., 2000).  $\square$

## 4 FINDING FAITHFUL POSETS

We now ask how to find faithful DAGs for simple marginal independence graphs. We observed that marginal independence graphs cannot distinguish between transitively equivalent DAGs, so a perhaps more natural question is: which posets are faithful to a given graph? As pointed out before, we can obtain all DAGs from faithful posets in a unique manner by removing transitive edges. A further advantage of the poset representation will turn out to be that the “smallest” and “largest” faithful posets can be characterized uniquely (up to isomorphism); as we shall also see, this is not as easy for DAGs, except for marginal independence graphs in a certain subclass.

### 4.1 Maximal Faithful Posets

Our first aim is to characterize the “upper bound” of the faithful set. That is, we wish to identify those posets for which no edge supergraph is also faithful. We will show that a construction described by Pearl and Wermuth (1994) solves exactly this problem.

**Definition 4.1.** For a graph  $\mathcal{U} = (V, E(\mathcal{U}))$ , the sink graph  $S(\mathcal{U}) = (V, E(S(\mathcal{U})))$  is constructed as follows: for each edge  $u - v$  in  $\mathcal{U}$ , add to  $E(S(\mathcal{U}))$ : (1) an edge  $u \rightarrow v$  if  $Bd(u) \subsetneq Bd(v)$ ; (2) an edge  $u \leftarrow v$  if  $Bd(u) \supsetneq Bd(v)$ ; (3) an edge  $u - v$  if  $Bd(u) = Bd(v)$ .

For instance, the sink graph of the graph in Figure 1a is the graph in Figure 1b.

**Definition 4.2** (Pearl and Wermuth (1994)). A sink orientation of a graph  $\mathcal{U}$  is any DAG obtained by replacing every undirected edge of  $S(\mathcal{U})$  by a directed edge.

We first need to state the following.

**Lemma 4.3.** *Every sink orientation of  $\mathcal{U}$  is a poset.*

*Proof.* Fix a sink orientation  $\mathcal{G}$  and consider any chain  $x \rightarrow y \rightarrow z$ . By construction, this implies that  $\text{Bd}(x) \subsetneq \text{Bd}(z)$ . Hence, if  $x$  and  $z$  are adjacent in the sink graph, then the only possible orientation is  $x \rightarrow z$ . There can be two reasons why  $x$  and  $z$  are not adjacent in the sink graph: (1) They are not adjacent in  $\mathcal{U}$ . But then  $\mathcal{G}$  would not be faithful, since  $\mathcal{G}$  implies the edge  $x - z$ . (2) The edge was not added to the sink graph. But this contradicts  $\text{Bd}(x) \subsetneq \text{Bd}(z)$ .  $\square$

This Lemma allows us to strengthen Theorem 2 by Pearl and Wermuth (1994) in the sense that we can replace “DAG” by “maximal poset” (emphasized):

**Theorem 4.4.**  *$\mathcal{P}$  is a maximal poset faithful to  $\mathcal{U}$  iff  $\mathcal{P}$  is a sink orientation of  $\mathcal{U}$ .*

The following is also not hard to see.

**Lemma 4.5.** *For a SMIG  $\mathcal{U}$ , every DAG  $\mathcal{G}$  that is faithful to  $\mathcal{U}$  is a subgraph of some sink orientation of  $\mathcal{U}$ .*

*Proof.* Obviously the skeleton of  $\mathcal{G}$  cannot contain edges that are not in  $\mathcal{U}$ . So, suppose  $x \rightarrow y$  is an edge in  $\mathcal{G}$  but conflicts with the sink orientation; that is, the sink graph contains the edge  $y \rightarrow x$ . That is the case only if  $\text{Bd}_{\mathcal{U}}(y)$  is a proper subset of  $\text{Bd}_{\mathcal{U}}(x)$ . However, in the marginal independence graph of  $\mathcal{G}$ , any node that is adjacent to  $x$  (has a common ancestor) must also be adjacent to  $y$ . Thus, the marginal independence graph of  $\mathcal{G}$  cannot be  $\mathcal{U}$ .  $\square$

Every maximal faithful poset for  $\mathcal{U}$  can be generated by first fixing a topological ordering of  $\mathcal{S}(\mathcal{U})$  and then generating the DAG that corresponds to that ordering, an idea that has also been mentioned by Drton and Richardson (2008a). This construction makes it obvious that all maximal faithful posets are isomorphic.

For curiosity of the reader, we note that  $\mathcal{S}(\mathcal{U})$  can also be viewed as a *complete partially directed acyclic graph* (CPDAG), which represents the Markov equivalence class of edge-maximal DAGs that are faithful with  $\mathcal{U}$ . CPDAGs are used in the context of inferring DAGs from data (Spirtes et al., 2000; Chickering, 2003; Kalisch and Bühlmann, 2007), which is only possible up to Markov equivalence.

## 4.2 Minimal Faithful Posets

A minimal faithful poset to  $\mathcal{U}$  is one from which no further relations can be deleted without entailing more independencies than are given by  $\mathcal{U}$ .

**Definition 4.6.** *Let  $\mathcal{U} = (V, E)$  be a graph and let  $I \subseteq V$  be an independent set. Then  $I_{\mathcal{U}}^{\rightarrow}$  is the poset consisting of the nodes in  $I$ , their neighbors in  $\mathcal{U}$ , and directed edges  $i \rightarrow j$  for each  $i, j$  where  $j \in N(i)$ .*

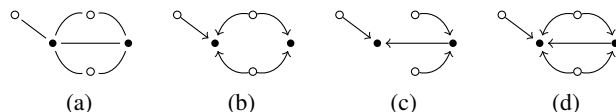


Figure 3: (a) A graph  $\mathcal{U}$  with three simplicial nodes  $I$  (open circles). (b) Its unique minimal faithful poset  $I_{\mathcal{U}}^{\rightarrow}$ . (c,d) The unique faithful DAGs with minimum (c) or maximum (d) numbers of edges.

For example, Figure 3b shows the unique  $I_{\mathcal{U}}^{\rightarrow}$  for the graph in Figure 3a.

**Theorem 4.7.** *Let  $\mathcal{U} = (V, E) \in \mathcal{U}$ . Then a poset  $\mathcal{P}$  is a minimal poset faithful to  $\mathcal{U}$  iff  $\mathcal{P} = I_{\mathcal{U}}^{\rightarrow}$  for a set  $I$  consisting of one simplicial vertex for each simplex.*

*Proof.* We first show that if  $I$  is a set consisting of one simplicial node for each simplex, then  $I_{\mathcal{U}}^{\rightarrow}$  is a minimal faithful poset. Every edge  $e \in E(\mathcal{U})$  resides in a simplex, so it is either adjacent to  $I$  or both of its endpoints are adjacent to some  $i \in I$ . In both cases,  $I_{\mathcal{U}}^{\rightarrow}$  implies  $e$ . Also  $I_{\mathcal{U}}^{\rightarrow}$  does not imply more edges than are in  $\mathcal{U}$ . Now, suppose we delete an edge  $i \rightarrow x$  from  $I_{\mathcal{U}}^{\rightarrow}$ . This edge must exist in  $\mathcal{U}$ , else  $i$  was not simplicial. But now  $I_{\mathcal{U}}^{\rightarrow}$  no longer implies this edge. Thus,  $I_{\mathcal{U}}^{\rightarrow}$  is minimal. Second, assume that  $\mathcal{P}$  is a minimal faithful poset. Assume  $\mathcal{P}$  would contain a sequence of two directed edges  $x \rightarrow y \rightarrow z$ . Then  $\mathcal{P}$  would also contain the edge  $x \rightarrow z$ . But then  $y \rightarrow z$  could be deleted from  $\mathcal{P}$  without changing the dependency graph, and  $\mathcal{P}$  was not minimal. So,  $\mathcal{P}$  does not contain any directed path of length more than 1. Next, observe that for each simplex in  $\mathcal{U}$ , the nodes must all have a common ancestor in  $\mathcal{P}$ . Without paths of length  $> 1$ , this is only possible if one node  $i$  in the simplex is a parent of all other nodes, and there are no edges among the child nodes of  $i$ . Finally, each such  $i$  must be a simplicial node in  $\mathcal{U}$ ; otherwise, it would reside in two or more simplexes, and would have to be the unique parent in those simplexes. But then the children of  $i$  would form a single simplex in  $\mathcal{U}$ .  $\square$

Like the maximal posets, all minimal posets are thus isomorphic. We point out that the minimal posets contain no transitive edges and therefore, they are also edge-minimal faithful DAGs. However, this does not imply that minimal posets have the smallest possible number of edges amongst all faithful DAGs (Figure 3). There appears to be no straightforward characterization of the DAGs with the smallest number of edges for marginal independence graphs in general. However, a beautiful one exists for the subclass of trivially perfect graphs.

**Definition 4.8.** *A tree poset is a poset whose transitive reduction is a tree (with edges pointing towards the root).*

**Theorem 4.9.** *A connected SMIG  $\mathcal{U}$  has a faithful tree poset iff it is trivially perfect.*



*Proof.* The bound graph of a tree poset is identical to its *comparability graph* (Brandstädt et al., 1999), which is the skeleton of the poset. Comparability graphs of tree posets coincide with trivially perfect graphs (Wolk, 1965).  $\square$

Since no connected graph on  $n$  nodes can have fewer edges than the transitive reduction of a tree poset on the same nodes (i.e.,  $n - 1$ ), tree posets coincide with faithful DAGs having the smallest possible number of edges.

How do we construct a tree for a given trivially perfect graph? Every such graph must have a *central point*, which is a node that is adjacent to all other nodes. We set this node as the sink of the tree, and continue recursively with the subgraphs obtained after removing the central point. Each subgraph is also trivially perfect and can thus be oriented into a tree. After we are done, we link the sinks of the trees of the subgraphs to the original central point to obtain the full tree (Wolk, 1965).

## 5 FINDING FAITHFUL DAGS

If a given marginal independence graph  $\mathcal{U}$  admits faithful DAG models, then it is of interest to enumerate these. A trivial enumeration procedure is the following: start with the sink graph of  $\mathcal{U}$ , choose an arbitrary edge  $e$ , and form all 2 or 3 subgraphs obtained by keeping  $e$  (if it is directed), orienting  $e$  (if it is undirected), or deleting it. Apply the procedure recursively to these subgraphs. During the recursion, do not touch edges that have been previously chosen. If the current graph is a DAG that is faithful to  $\mathcal{U}$ , output it; otherwise, stop the recursion.

However, we can do better by exploiting the results of the previous section, which will allow us to derive enumeration algorithms that generate representations of multiple DAGs at each step.

### 5.1 Enumeration of Faithful DAGs

Having characterized the maximal and minimal faithful posets, we are now ready to construct an enumeration procedure for all DAGs that are faithful to a given graph. We first state the following combination of Theorem 4.4 and Theorem 4.7.

**Proposition 5.1.** A DAG  $\mathcal{G} = (V, E(\mathcal{G}))$  is faithful to a SMIG  $\mathcal{U} = (V, E(\mathcal{U}))$  iff (1)  $\mathcal{G}$  is an edge subgraph of some sink orientation of  $\mathcal{U}$  and (2) the transitive closure of  $\mathcal{G}$  is an edge supergraph of  $I_{\mathcal{U}}^{\rightarrow}$  for some node set  $I$  consisting of one simplicial node for each simplex.

From this observation, we can derive our first construction procedure for faithful DAGs.

**Proposition 5.2.** A DAG  $\mathcal{G}$  is faithful to a SMIG  $\mathcal{U} = (V, E(\mathcal{U}))$  iff it can be generated by the following steps. (1) Pick any set  $I \subseteq V$  consisting of one simplicial node

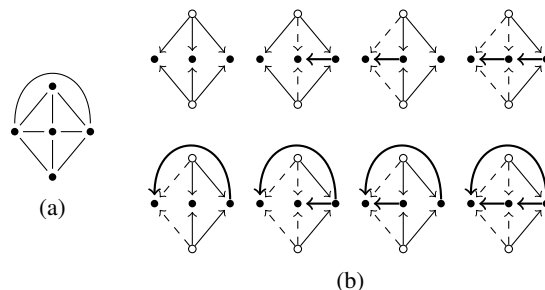


Figure 4: Example of the procedure in Proposition 5.2 that, given a SMIG (a), enumerates all faithful DAGs (b). For brevity, only the graphs that correspond to a fixed topological ordering are displayed. Only one set  $I$  (open circles) can be chosen in step (1). Thick edges and filled nodes highlight the DAG  $\mathcal{G}$ . Mandatory edges (solid) link  $I$  to the sources of  $\mathcal{G}$ ; if any such edge was absent, one of the relationships in the poset  $I_{\mathcal{U}}^{\rightarrow}$  would be missing. Optional edges (dashed) are transitively implied from the mandatory ones and  $\mathcal{G}$ .

for each simplex. (2) Generate any DAG on the nodes  $V \setminus I$  that is an edge subgraph of some sink orientation of  $\mathcal{U}$ . (3) Add any subset of edges from  $I_{\mathcal{U}}^{\rightarrow}$  such that the transitive closure of the resulting graph contains all edges of  $I_{\mathcal{U}}^{\rightarrow}$ .

While step (3) may seem ambiguous, Figure 4 illustrates that after step (2), the edges from  $I_{\mathcal{U}}^{\rightarrow}$  decompose nicely into *mandatory* and *optional* ones. This means that we can in fact stop the construction procedure after step (2) and output a “graph pattern”, in which some edges are marked as optional. This is helpful in light of the potentially huge space of faithful models, because every graph pattern can represent an exponential number of DAGs.

### 5.2 Enumeration of Faithful Posets

The DAGs resulting from the procedure in Proposition 5.2 are in general redundant because no care is taken to avoid generating transitive edges. By combining Propositions 5.1 and 5.2, we obtain an algorithm that generates sparse, non-redundant representations of the faithful DAGs.

**Theorem 5.3.** A poset  $\mathcal{P}$  is faithful to  $\mathcal{U} = (V, E(\mathcal{U}))$  iff it can be generated by the following steps. (1) Pick any set  $I \subseteq V$  consisting of one simplicial node for each simplex. (2) Generate a poset  $\mathcal{P}$  on the nodes  $V \setminus I$  that is an edge subgraph of some sink orientation of  $\mathcal{U}$ . (3) Add  $I_{\mathcal{U}}^{\rightarrow}$  to  $\mathcal{P}$ .

A nice feature of this construction is that step (3) is unambiguous: every choice for  $I$  in step (1) and  $\mathcal{P}$  in step (2) yields exactly one poset. Figure 5 gives an explicit pseudocode for an algorithm that uses Theorem 5.3 to enumerate all faithful posets.

Our algorithm is efficient in the sense that at every inter-

```

function FAITHFULPOSETS( $\mathcal{U} = (V(\mathcal{U}), E(\mathcal{U}))$ )
  function LISTPOSETS( $\mathcal{G}, \mathcal{S}, R, I_{\mathcal{U}}^{\rightarrow}$ )
    if  $\mathcal{G}$  is acyclic and atransitive then
      Output  $\mathcal{G} \cup I_{\mathcal{U}}^{\rightarrow}$ 
      if skeleton of  $\mathcal{G} \subsetneq$  skeleton of  $\mathcal{S}$  then
         $e \leftarrow$  some edge consistent with  $E(\mathcal{S}) \setminus R$ 
        LISTPOSETS( $\mathcal{G}, \mathcal{S}, R \cup \{e\}, I_{\mathcal{U}}^{\rightarrow}$ )
         $E(\mathcal{G}) \leftarrow E(\mathcal{G}) \cup \{e\}$ 
        LISTPOSETS( $\mathcal{G}, \mathcal{S}, R \cup \{e\}, I_{\mathcal{U}}^{\rightarrow}$ )
  for all node sets  $I$  of  $\mathcal{U}$  consisting of one simplicial
  node per simplex do
     $\mathcal{G} \leftarrow$  empty graph on nodes of  $V(\mathcal{U}) \setminus I$ 
     $\mathcal{S} \leftarrow$  sink graph of  $\mathcal{U}$  on nodes of  $V(\mathcal{U}) \setminus I$ 
    LISTPOSETS( $\mathcal{G}, \mathcal{S}, \emptyset, I_{\mathcal{U}}^{\rightarrow}$ )

```

Figure 5: Enumeration algorithm for faithful posets.

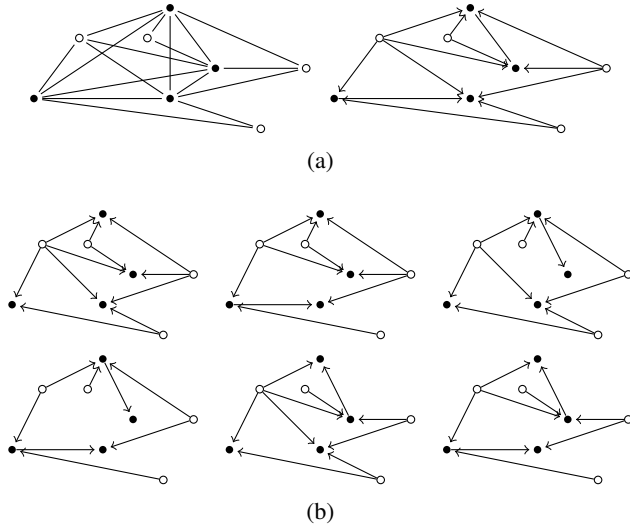


Figure 6: (a) A graph  $\mathcal{U}$  and its sink graph. (b) Transitive reductions of all 6 faithful posets that are generated by Algorithm FAITHFULPOSETS for the input graph (a).

nal node in its recursion tree, it outputs a faithful poset. At every node we need to evaluate whether the current  $\mathcal{G}$  is acyclic and atransitive (i.e., contains no transitive edges), which can be done in polynomial time. Also simplexes and their simplicial vertices can be found in polynomial time Kloks et al. (2000). Thus, our algorithm is a *polynomial delay enumeration algorithm* similar to the ones used to enumerate adjustment sets for DAGs (Textor and Liškiewicz, 2011; van der Zander et al., 2014). Figure 6 shows an example output for this algorithm.

## 6 EXAMPLE APPLICATIONS

In this section, we apply the previous results to explore some explicit combinatorial properties of SMIGs and their faithful DAGs.

| $n$ | connected graphs | conn. SMIGs | unique DAG |
|-----|------------------|-------------|------------|
| 2   | 1                | 1           | 0          |
| 3   | 2                | 2           | 1          |
| 4   | 6                | 4           | 1          |
| 5   | 21               | 10          | 2          |
| 6   | 112              | 27          | 4          |
| 7   | 853              | 88          | 10         |
| 8   | 11,117           | 328         | 27         |
| 9   | 261,080          | 1,460       | 90         |
| 10  | 11,716,571       | 7,799       | 366        |

Table 1: Comparison of the number of unlabeled connected graphs with  $n$  nodes to the number of such graphs that are also SMIGs. For  $n = 13$  (not shown), non-SMIGs outnumber SMIGs by more than  $10^7 : 1$ .

### 6.1 Counting SMIGs

We revisit the question: when can a marginal independence graph allow a causal interpretation (Pearl and Wermuth, 1994)? More precisely, we ask *how many* marginal independence graphs on  $n$  variables are SMIGs. We reformulate this question into a version that has been investigated in the context of poset theory. Let the *height* of a poset  $\mathcal{P}$  be the length of a longest path in  $\mathcal{P}$ . The following is an obvious implication of Theorem 4.7.

**Corollary 6.1.** *The number  $M(n)$  of non-isomorphic SMIGs with  $n$  nodes is equal to the number of non-isomorphic posets on  $n$  variables of height 1.*

Enumeration of posets is a highly nontrivial problem, and an intensively studied one. The online encyclopedia of integer sequences (OEIS) tabulates  $M(n)$  for  $n$  up to 40 (Wambach, 2015). We give the first 10 entries of the sequence in Table 1 and compare it to the number of graphs in general (up to isomorphism). As we observe, the fraction of graphs that admit a DAG on the same variables decreases swiftly as  $n$  increases.

### 6.2 Graphs with a Unique Faithful DAG

From a causal inference viewpoint, the best we can hope for is a SMIG to which only single, unique DAG is faithful. The classical example is the graph  $\cdot - \cdot - \cdot$ , which for more than 3 nodes generalizes to a “star” graph. However, for 5 or more nodes there are graphs other than the star which also induce a single unique DAG. Combining Lemma 4.5 and Theorem 4.7 allows for a simple characterization of all such SMIGs.

**Corollary 6.2.** *A SMIG  $\mathcal{U}$  with  $n$  nodes has a unique faithful DAG iff each of its simplexes contains only one simplicial node and its sink orientation equals  $I_{\mathcal{U}}^{\rightarrow}$ .*

Based on this characterization, we computed the number of

| $n$ | posets with $n$ nodes | faithful to $C_n$ |
|-----|-----------------------|-------------------|
| 1   | 1                     | 1                 |
| 2   | 3                     | 2                 |
| 3   | 19                    | 9                 |
| 4   | 219                   | 76                |
| 5   | 4,231                 | 1,095             |
| 6   | 130,023               | 25,386            |
| 7   | 6,129,859             | 910,161           |
| 8   | 431,723,379           | 49,038,872        |
| 9   | 44,511,042,511        | 3,885,510,411     |
| 10  | 6,611,065,248,783     | 445,110,425,110   |

Table 2: Possible labelled posets on  $n$  variables before and after observing a complete SMIG  $C_n$ .

SMIGs with unique DAGs for  $n$  up till 9 (Table 1). Interestingly, this integer sequence does not seem to correspond to any known one.

### 6.3 Information Content of a SMIG

How much information does a marginal independence graph contain? Let us denote the number of posets on  $n$  variables by  $P(n)$ . After observing a marginal independence graph  $\mathcal{U}$ , the number of models that are still faithful to the data reduces to size  $P(n) - k(\mathcal{U})$ , where  $k(\mathcal{U}) \leq P(n)$  (indeed, quite often  $k(\mathcal{U}) = P(n)$  as we can see in Table 1). Of course, the number  $k(\mathcal{U})$  strongly depends on the structure of the SMIG  $\mathcal{U}$ . But even in the worst case when  $\mathcal{U}$  is a complete graph, the space of possible models is still reduced because not all DAGs entail a complete marginal independence graph.

Thus, the following simple consequence of Theorem 4.7 helps to derive a worst-case bound on how much a SMIG reduces structural uncertainty with respect to the model space of posets with  $n$  variables.

**Corollary 6.3.** *The number of faithful posets with respect to a complete graph with  $n$  nodes is  $n$  times the number of posets with  $n - 1$  nodes.*

Table 2 lists the number of possible posets before and after observing a complete SMIG for up to 10 variables. In this sense, at  $n = 10$ , the uncertainty is reduced about 15-fold.

We note that a similar but more technical analysis is possible for uncertainty reduction with respect to DAGs instead of posets. We omit this due to space limitations.

## 7 MODELS WITH LATENT VARIABLES

In this section we consider situations in which a graph  $\mathcal{U}$  is not a SMIG (which can be detected using the algorithm in Theorem 3.13). Similarly to the definition proposed in Pearl and Verma (1987) for the general dependency models, to obtain faithful DAGs for such graphs we will extend

the DAGs with some auxiliary nodes. We generalize Definition 3.1 as follows.

**Definition 7.1.** *Let  $\mathcal{U} = (V, E(\mathcal{U}))$  be a graph and let  $Q$ , with  $Q \cap V = \emptyset$ , be a set of auxiliary nodes. A DAG  $\mathcal{G} = (V \cup Q, E(\mathcal{G}))$  is faithful to  $\mathcal{U}$  if for all  $v, w \in V$ ,  $v - w \in E(\mathcal{U})$  iff  $v$  and  $w$  have a common ancestor in  $\mathcal{G}$ .*

The result below follows immediately from Proposition 3.12.

**Proposition 7.2.** *For every graph  $\mathcal{U}$  there exists a faithful DAG  $\mathcal{U}$  with some auxiliary nodes.*

Obviously, if  $\mathcal{U} \in \text{SMIG}$  then there exists a faithful DAG to  $\mathcal{U}$  with  $Q = \emptyset$ . For  $\mathcal{U} \notin \text{SMIG}$ , from the proof of Proposition 3.12 it follows that there exists a set  $Q$  of at most  $|E(\mathcal{U})|$  nodes and a DAG  $\mathcal{G}$  such that  $\mathcal{G}$  is faithful to  $\mathcal{U}$  with auxiliary nodes  $Q$ . But the problem arises to minimize the cardinality of  $Q$ .

**Theorem 7.3.** *The problem to decide if for a given graph  $\mathcal{U}$  and an integer  $k$ , there exists a faithful DAG with at most  $k$  auxiliary nodes, is NP-complete.*

*Proof.* It is easy to see that the problem is in NP. To prove that it is NP-hard, we show a polynomial time reduction from the edge clique cover problem, that is known to be NP-complete (Karp, 1972). Recall that the problem edge clique cover is to decide if for a graph  $\mathcal{U}$  and an integer  $k$  there exist a set of  $k$  subgraphs of  $\mathcal{U}$ , such that each subgraph is a clique and each edge of  $\mathcal{U}$  is contained in at least one of these subgraphs?

Let  $\mathcal{U} = (V, E)$  and  $k$  be an instance of the edge clique cover problem, with  $V = \{v_1, \dots, v_n\}$ . We construct the marginal independence graph  $\mathcal{U}'$  as follows. Let  $W = \{w_1, \dots, w_n\}$ . Then  $V(\mathcal{U}') = V \cup W$  and  $E(\mathcal{U}') = E \cup \{v_i - w_i : i = 1, \dots, n\}$ . Obviously,  $\mathcal{U}'$  can be constructed from  $\mathcal{U}$  in polynomial time. We claim that  $\mathcal{U} = (V, E)$  can be covered by  $\leq k$  cliques iff for  $\mathcal{U}'$  there exists a faithful DAG  $\mathcal{G}$  with at most  $k$  auxiliary nodes.

Assume first that  $\mathcal{U} = (V, E)$  can be covered by at most  $k$  cliques, let us say  $C_1, \dots, C_{k'}$ , with  $k' \leq k$ . Then we can construct a faithful DAG  $\mathcal{G}$  for  $\mathcal{U}'$  with  $k'$  auxiliary nodes as follows. Its set of nodes is  $V(\mathcal{G}) = V \cup W \cup Q$ , where  $Q = \{q_1, \dots, q_{k'}\}$ . The edges  $E(\mathcal{G})$  can be defined as

$$\{w_i \rightarrow v_i : i = 1, \dots, n\} \cup \bigcup_j \{q_j \rightarrow v : v \in C_j\}.$$

It is easy to see that  $\mathcal{G}$  is faithful to  $\mathcal{U}'$ .

Now assume that a DAG  $\mathcal{G}$ , with at most  $k$  auxiliary nodes  $Q$ , is faithful to  $\mathcal{U}'$ . From the construction of  $\mathcal{U}'$  it follows that for all different nodes  $v_i, v_j \in V$  there is no directed path from  $v_i$  to  $v_j$  in  $\mathcal{G}$ . If such a path exists, then  $v_i$  is an ancestor of  $v_j$  in  $\mathcal{G}$ . Since  $v_i - w_i$  is an edge of  $\mathcal{U}'$ , the nodes  $v_i$  and  $w_i$  have a common ancestor in  $\mathcal{G}$ , which must be also

a common ancestor of  $w_i$  and  $v_j$  – a contradiction because  $w_i$  and  $v_j$  are not incident in  $\mathcal{U}'$ . Thus, all treks connecting pairs of nodes from  $V$  in  $\mathcal{G}$  must contain auxiliary nodes.

Next, we slightly modify  $\mathcal{G}$ : for each  $w_i$  we remove all incident edges and add the new edge  $w_i \rightarrow v_i$ . The resulting graph  $\mathcal{G}'$ , is a DAG which remains faithful to  $\mathcal{U}'$ . Indeed, we cannot obtain a directed cycle in the  $\mathcal{G}'$  since no  $w_i$  has an in-edge and the original  $\mathcal{G}$  was a DAG. To see that the obtained DAG remains faithful to  $\mathcal{U}'$  note first that after the modifications,  $w_i$  and  $v_i$  have a common ancestor in  $\mathcal{G}$  whereas  $w_i$  and  $v_j$ , with  $i \neq j$ , do not. Otherwise, it would imply a directed path from  $v_i$  to  $v_j$  since  $w_i$  is the only possible ancestor of both nodes – a contradiction. Finally, note that any trek connecting  $v_i$  and  $v_j$  in  $\mathcal{G}$  cannot contain a node from  $W$ . Similarly, no trek between  $v_i$  and  $v_j$  in  $\mathcal{G}'$  contains a node from  $W$ . We get that  $v_i$  and  $v_j$  have a common ancestor in  $\mathcal{G}$  iff they have a common ancestor in  $\mathcal{G}'$ .

Thus, in  $\mathcal{G}'$  the auxiliary nodes  $Q$  are incident to  $V$ , but not to nodes from  $W$ . Below we modify  $\mathcal{G}'$  further and obtain a DAG  $\mathcal{G}''$ , in which every auxiliary node is incident with a node in  $V$  via an out-edge only. To this aim we remove from  $\mathcal{G}'$  all edges going out from a node in  $V$  to a node in  $Q$ .

Obviously, if  $v_i$  and  $v_j$  have a common ancestor in  $\mathcal{G}''$ , then they also have a common ancestor in  $\mathcal{G}'$ , because  $E(\mathcal{G}'') \subseteq E(\mathcal{G}')$ . The opposite direction follows from the fact we have shown at the beginning of this proof that for all different nodes  $v_i, v_j \in V$  there is no directed path from  $v_i$  to  $v_j$  in  $\mathcal{G}$ . This is true also for  $\mathcal{G}'$ . Thus, if  $v_i$  and  $v_j$  have a common ancestor, say  $x$ , in  $\mathcal{G}'$  then  $x \in Q$  and there exist directed paths  $x \rightarrow y_1 \rightarrow \dots \rightarrow y_r \rightarrow v_i$  and  $x \rightarrow y'_1 \rightarrow \dots \rightarrow y'_{r'} \rightarrow v_j$  such that also all  $y_1, \dots, y_r$  and  $y'_1, \dots, y'_{r'}$  belong to  $Q$ . But from the construction of  $\mathcal{G}''$  it follows that both paths belong also to  $\mathcal{G}''$ .

Since  $\mathcal{G}''$  is faithful to  $\mathcal{U}$ , for every auxiliary node  $Q$  the subgraph induced by its children  $Ch(Q) \cap V$  in  $\mathcal{G}''$  is a clique in  $\mathcal{U}'$ . Moreover every edge  $v_i - v_j$  of the graph  $\mathcal{U}$  belongs to at least one such clique. Thus the subgraphs induced by  $Ch(q_1) \cap V, \dots, Ch(q_{k'}) \cap V$ , with  $k' \leq k$ , are cliques that cover  $\mathcal{U}$ .  $\square$

## 8 DISCUSSION

Given a graph that represents a set of pairwise marginal independencies, which causal structures on the same variables might have generated this graph? Here we characterized all these structures, or alternatively, all maximal and minimal ones. Furthermore, we have shown that it is possible to deduce how many exogenous variables (which correspond to simplicial nodes) the causal structure might have, and even to tell whether it might be a tree. For graphs that do not admit a DAG on the same variables, we have studied

the problem of explaining the data with as few additional variables as possible, and proved it to be NP-hard. This may be surprising; the related problem of finding a mixed graph that is Markov equivalent to a bidirected graph and has as few bidirected edges as possible is efficiently solvable (Drton and Richardson, 2008a).

The connection to posets emphasizes that sets of faithful DAGs have complex combinatorics. Indeed, if there are no pairwise independent variables, then we obtain the classical poset enumeration problem (Brinkmann and McKay, 2002). Our current, unoptimized implementation of the algorithm in Figure 5 allows us to deal with dense graphs up to about 12 nodes (sparse graphs are easier to deal with). We point out that our enumeration algorithms operate with a “template graph”, i.e., the sink orientation. It is possible to incorporate certain kinds of background knowledge, like a time-ordering of the variables, into this template graph by deleting some edges. Such further constraints could greatly reduce the search space. Another additional constraint that could be used for linear models is the precision matrix (Cox and Wermuth, 1993; Pearl and Wermuth, 1994), though finding DAGs that explain a given precision matrix is NP-hard in general (Verma and Pearl, 1993),

We observed that the pairwise marginal independencies substantially reduce structural uncertainty even in the worst case (Table 1). Causal inference algorithms could exploit this to reduce the number of CI tests. The PC algorithm (Kalisch and Bühlmann, 2007), for instance, forms the marginal independence graph as a first stage before performing any CI tests. At that stage, it could be immediately tested if the resulting graph is a SMIG, and if not, the algorithm can terminate as no faithful DAG exists.

In summary, we have mapped out the space of causal structures that are faithful to a given set of pairwise marginal independencies using constructive criteria that lead to well-structured enumeration procedures. The central idea underlying our results is that faithful models for marginal independencies are better described by posets than by DAGs. Our results allow to quantify how much our uncertainty about a causal structure is reduced when we invoke the faithfulness assumption and observe a set of marginal independencies.

In future work, it would be interesting to extend our approach to small (instead of empty) conditioning sets, which would cover cases where we only wish to perform CI tests with low dimensionality.

## References

- W. P. Bergsma. Testing conditional independence for continuous random variables. Technical Report 2004-049, EURANDOM, 2004.
- A. Brandstädt, J. P. Spinrad, et al. *Graph classes: a survey*, volume 3. Siam, 1999.
- G. Brinkmann and B. D. McKay. Posets on up to 16 points. *Order*, 19(2):147–179, 2002.
- G. A. Cheston and T. Jap. A survey of the algorithmic properties of simplicial, upper bound and middle graphs. *Journal of Graph Algorithms and Applications*, 10(2):159–190, 2006.
- D. M. Chickering. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3:507–554, 2003.
- D. Colombo, M. H. Maathuis, M. Kalisch, and T. S. Richardson. Learning high-dimensional directed acyclic graphs with latent and selection variables. *Annals of Statistics*, 40(1):294–321, 2012.
- D. R. Cox and N. Wermuth. Linear dependencies represented by chain graphs. *Statistical Science*, 8(3):204–283, 1993.
- G. Doran, K. Muandet, K. Zhang, and B. Schölkopf. A permutation-based kernel conditional independence test. In *Proceedings of UAI 2014*, pages 132–141, 2014.
- M. Drton and T. S. Richardson. A new algorithm for maximum likelihood estimation in gaussian graphical models for marginal independence. In *Proceedings of UAI 2003*, pages 184–191, 2003.
- M. Drton and T. S. Richardson. Graphical methods for efficient likelihood inference in gaussian covariance models. *Journal of Machine Learning Research*, 9:893–914, 2008a.
- M. Drton and T. S. Richardson. Binary models for marginal independence. *Journal of the Royal Statistical Society, Ser. B*, 70(2):287–309, 2008b.
- A. Idelberger. Generating causal diagrams from stochastic dependencies (in German). Master’s thesis, Universität zu Lübeck, Germany, 2014.
- M. Kalisch and P. Bühlmann. Estimating high-dimensional directed acyclic graphs with the PC-algorithm. *Journal of Machine Learning Research*, 8:613–636, 2007.
- R. M. Karp. *Reducibility among combinatorial problems*. Springer, 1972.
- T. Kloks, D. Kratsch, and H. Müller. Finding and counting small induced subgraphs efficiently. *Information Processing Letters*, 74:115–121, 2000.
- F. McMorris and T. Zaslavsky. Bound graphs of a partially ordered set. *Journal of Combinatorics, Information & System Sciences*, 7:134–138, 1982. ISSN 0250-9628; 0976-3473/e.
- J. Pearl and T. Verma. The logic of representing dependencies by directed graphs. In *Proceedings of AAAI 1987 – Volume 1*, pages 374–379. AAAI Press, 1987.
- J. Pearl and N. Wermuth. *When Can Association Graphs Admit A Causal Interpretation?*, volume 89 of *Lecture Notes in Statistics*, pages 205–214. Springer, 1994.
- J. M. Peña. Reading dependencies from covariance graphs. *International Journal of Approximate Reasoning*, 54(1):216–227, 2013.
- T. S. Richardson. Markov properties for acyclic directed mixed graphs. *The Scandinavian Journal of Statistics*, 30(1):145–157, 2003.
- M. Skowrońska and M. M. Sysło. An algorithm to recognize a middle graph. *Discrete Applied Mathematics*, 7(2):201–208, 1984. ISSN 0166-218X.
- P. Spirtes, C. N. Glymour, and R. Scheines. *Causation, prediction, and search*. MIT press, 2000.
- K. M. Tan, P. London, K. Mohan, S.-I. Lee, M. Fazel, and D. Witten. Learning Graphical Models With Hubs. *Journal of Machine Learning Research*, 15:3297–3331, Oct 2014.
- J. Textor and M. Liškiewicz. Adjustment criteria in causal diagrams: An algorithmic perspective. In *Proceedings of UAI 2011*, pages 681–688. AUAI Press, 2011.
- B. van der Zander, M. Liškiewicz, and J. Textor. Constructing separators and adjustment sets in ancestral graphs. In *Proceedings of UAI 2014*, pages 907–916, 2014.
- T. Verma and J. Pearl. Deciding morality of graphs is NP-complete. In *Proceedings of UAI 1993*, pages 391–399, 1993.
- G. Wambach. The On-Line Encyclopedia of Integer Sequences. <http://oeis.org/A007776>, 2015. Number of connected posets with  $n$  elements of height 1. Accessed in March 2015.
- E. S. Wolk. A note on the comparability graph of a tree. *Proceedings of the American Mathematical Society*, 16:17–20, 1965.
- K. Zhang, J. Peters, D. Janzing, and B. Schölkopf. Kernel-based conditional independence test and application in causal discovery. In *Proceedings of UAI 2011*, pages 804–8013, 2011.

---

# Bethe Projections for Non-Local Inference

---

**Luke Vilnis\***  
UMass Amherst  
luke@cs.umass.edu

**David Belanger\***  
UMass Amherst  
belanger@cs.umass.edu

**Daniel Sheldon**  
UMass Amherst  
sheldon@cs.umass.edu

**Andrew McCallum**  
UMass Amherst  
mccallum@cs.umass.edu

## Abstract

Many inference problems in structured prediction are naturally solved by augmenting a tractable dependency structure with complex, non-local auxiliary objectives. This includes the mean field family of variational inference algorithms, soft- or hard-constrained inference using Lagrangian relaxation or linear programming, collective graphical models, and forms of semi-supervised learning such as posterior regularization. We present a method to discriminatively *learn* broad families of inference objectives, capturing powerful non-local statistics of the latent variables, while maintaining tractable and provably fast inference using non-Euclidean projected gradient descent with a distance-generating function given by the Bethe entropy. We demonstrate the performance and flexibility of our method by (1) extracting structured citations from research papers by learning soft global constraints, (2) achieving state-of-the-art results on a widely-used handwriting recognition task using a novel learned non-convex inference procedure, and (3) providing a fast and highly scalable algorithm for the challenging problem of inference in a collective graphical model applied to bird migration.

## 1 INTRODUCTION

Structured prediction has shown great success in modeling problems with complex dependencies between output variables. Practitioners often use undirected graphical models, which encode conditional dependency relationships via a graph. However, the tractability of exact inference in these models is limited by the graph’s *treewidth*, often yielding a harsh tradeoff between model expressivity and tractability.

---

\* Equal contribution.

Graphical models are good at modeling local dependencies between variables, such as the importance of surrounding context in determining the meaning of words or phrases. However, their sensitivity to cyclic dependencies often renders them unsuitable for modeling preferences for certain globally consistent states. For example, in the canonical NLP task of part-of-speech tagging, there is no clear way to enforce the constraint that every sentence have at least one verb without increasing the likelihood that *every* token is predicted to be a verb.

Concretely, exact marginal inference in a discrete graphical model can be posed as the following optimization problem

$$\boldsymbol{\mu}^* = \arg \min_{\boldsymbol{\mu} \in \mathcal{M}} -H(\boldsymbol{\mu}) - \langle \boldsymbol{\theta}, \boldsymbol{\mu} \rangle, \quad (1)$$

where  $\boldsymbol{\mu}$  is a concatenated vector of node and clique marginals,  $H(\boldsymbol{\mu})$  is the entropy,  $\mathcal{M}$  is the marginal polytope, and  $\boldsymbol{\theta}$  are parameters. Here we face a tradeoff: adding long-range dependencies directly to the model increases the clique size and thus the complexity of the problem and size of  $\boldsymbol{\mu}$ , rendering inference intractable. However, the linear scoring function  $\boldsymbol{\theta}$  breaks down over cliques, preventing us from enforcing global regularities in any other way. In this work, we propose to augment the inference objective (1) and instead optimize

$$\boldsymbol{\mu}^* = \arg \min_{\boldsymbol{\mu} \in \mathcal{M}} -H(\boldsymbol{\mu}) - \langle \boldsymbol{\theta}, \boldsymbol{\mu} \rangle + L_{\psi}(\boldsymbol{\mu}). \quad (2)$$

Here,  $L_{\psi}$  is some arbitrary parametric function of the entire concatenated marginal vector, where  $\psi$  may depend on input features. Since  $L_{\psi}$  is non-linear, it can enforce many types of non-local properties. Interestingly, whenever  $L_{\psi}$  is convex, and whenever inference is easy in the underlying model, i.e., solving (1) is tractable, we can solve (2) using non-Euclidean projected gradient methods using the Bethe entropy as a distance-generating function. Unlike many message-passing algorithms, our procedure maintains primal feasibility across iterations, allowing its use as an *any-time* algorithm. Furthermore, for non-convex  $L_{\psi}$ , we also show convergence to a local optimum of (2). Finally, we

present algorithms for discriminative learning of the parameters  $\psi$ . In a slight abuse of terminology, we call  $L_\psi$  a *non-local energy function*.

Ours is not the first work to consider modeling global preferences by augmenting a tractable base inference objective with non-local terms. For example, generalized mean-field variational inference algorithms augment a tractable distribution (the  $Q$  distribution) with a non-linear, non-convex global energy function that scores terms in the full model (the  $P$  distribution) using products of marginals of  $Q$  (Wainwright & Jordan, 2008). This is one special case of our non-local inference framework, and we present algorithms for solving the problem for much more general  $L_\psi$ , with compelling applications.

Additionally, the modeling utility provided by global preferences has motivated work in *dual decomposition*, where inference in loopy or globally-constrained models is decomposed into repeated calls to inference in tractable independent subproblems (Komodakis et al., 2007; Sontag et al., 2011). It has seen wide success due to its ease of implementation, since it reuses existing inference routines as black boxes. However, the technique is restricted to modeling linear constraints, imposed *a priori*. Similarly, these types of constraints have also been imposed on expectations of the posterior distribution for use in semi-supervised learning, as in *posterior regularization* and *generalized expectation* (Ganchev et al., 2010; Mann & McCallum, 2010). In contrast, our methods are designed to discriminatively learn expressive inference procedures, with minimal domain knowledge required, rather than regularizing inference and learning.

First, we provide efficient algorithms for solving the marginal inference problem (2) and performing MAP prediction in the associated distribution, for both convex and non-convex global energy functions. After that, we provide a learning algorithm for  $\theta$  and the parametrized  $L_\psi$  functions using an interpretation of (2) as approximate variational inference in a probabilistic model. All of our algorithms are easy to implement and rely on simple wrappers around black-box inference subroutines.

Our experiments demonstrate the power and generality of our approach by achieving state-of-the-art results on several tasks. We extract accurate citations from research papers by learning discriminative global regularities of valid outputs, outperforming a strong dual decomposition-based baseline (Anzaroot et al., 2014). In a benchmark OCR task (Taskar et al., 2004), we achieve state-of-the-art results with a learned non-convex, non-local energy function, that guides output decodings to lie near dictionary words. Finally, our general algorithm for solving (2) provides large speed improvements for the challenging task of inference in chain-structured *collective graphical models* (CGMs), applied to bird migration (Sheldon & Dietterich, 2011).

## 2 BACKGROUND

Let  $\mathbf{y} = (y_1, \dots, y_n)$  denote a set of discrete variables and  $\mathbf{x}$  be a collection of input features. We define the conditional distribution  $P_\theta(\mathbf{y}|\mathbf{x}) = \exp(\langle \theta(\mathbf{x}), S(\mathbf{y}) \rangle) / Z$ , where  $S(\mathbf{y})$  is a mapping from  $\mathbf{y}$  to a set of sufficient statistics,  $\theta(\mathbf{x})$  is a differentiable vector-valued mapping, and  $Z = \sum_{\mathbf{y}} \exp(\langle \theta, S(\mathbf{y}) \rangle)$ . Conditional random fields (CRFs) assume that  $(y_1, \dots, y_n)$  are given a graph structure and  $S(\mathbf{y})$  maps  $\mathbf{y}$  to a 0-1 vector capturing joint settings of each clique (Lafferty et al., 2001). Going forward, we often suppress the explicit dependency of  $\theta$  on  $\mathbf{x}$ . For fixed  $\theta$ , the model is called a Markov random field (MRF).

Given a distribution  $P(\mathbf{y})$ , define the expected sufficient statistics operator  $\mu(P) = \mathbb{E}_P[S(\mathbf{y})]$ . For the CRF statistics  $S(\mathbf{y})$  above,  $\mu$  is a concatenated vector of node and clique marginals. Therefore, *marginal inference*, the task of finding the marginal distribution of  $P_\theta(\mathbf{y}|\mathbf{x})$  over  $\mathbf{y}$ , is equivalent to computing the expectation  $\mu(P_\theta(\mathbf{y}|\mathbf{x}))$ .

For tree-structured graphical models,  $P_\theta(\mathbf{y}|\mathbf{x}) \leftrightarrow \mu(P_\theta(\mathbf{y}|\mathbf{x}))$  is a bijection, though this is not true for general graphs. Furthermore, for trees the entropy  $H(P_\theta(\mathbf{y}|\mathbf{x}))$  is equal to the Bethe entropy  $H_B(\mu(P_\theta(\mathbf{y}|\mathbf{x})))$ , defined, for example, in Wainwright & Jordan (2008). The *marginal polytope*  $\mathcal{M}$  is the set of  $\mu$  that correspond to some  $P_\theta$ .

As mentioned in the introduction, marginal inference can be posed as the optimization problem (1). MAP inference finds the joint setting  $\mathbf{y}$  with maximum probability. For CRFs, this is equivalent to

$$\arg \min_{\mathbf{y}} \langle -\theta(\mathbf{x}), S(\mathbf{y}) \rangle. \quad (3)$$

For tree-structured CRFs, marginal and MAP inference can be performed efficiently using dynamic programming. Our experiments focus on such graphs. However, the inference algorithms we present can be extended to general graphs wherever marginal inference is tractable using a convex entropy approximation and a local polytope relaxation.

## 3 MARGINAL INFERENCE WITH NON-LOCAL ENERGIES

We move beyond the standard inference objective (1), augmenting it with a non-local energy term as in (2):

$$\mu^* = \arg \min_{\mu \in \mathcal{M}} -H_B(\mu) - \langle \theta, \mu \rangle + L_\psi(\mu).$$

Here,  $L_\psi$  is some arbitrary parametrized function of the marginals, and  $\psi$  may depend on input features  $\mathbf{x}$ .

Intuitively, we are augmenting the inference objective (1) by allowing it to optimize a broader set of tradeoffs – not

only between expected node scores, clique scores, and entropy, but also global functions of the marginals. To be concrete, in our citation extraction experiments (Section 8.1), for example, we employ the simple structure:

$$L_\psi(\boldsymbol{\mu}) = \sum_j \psi_j \ell_j(\boldsymbol{\mu}), \quad (4)$$

Where each  $\ell_j$  is a univariate convex function and each  $\psi_j$  is constrained to be non-negative, in order to maintain the overall convexity of  $L_\psi$ . We further employ

$$\ell_j(\boldsymbol{\mu}) = \tilde{\ell}_j(a_j^\top \boldsymbol{\mu}), \quad (5)$$

where  $a_j$  encodes a ‘linear measurement’ of the marginals and  $\tilde{\ell}_j$  is some univariate convex function.

#### 4 VARIATIONAL INTERPRETATION AND MAP PREDICTION

We next provide two complementary interpretations of (2) as variational inference in a class of tractable probability distributions over  $\mathbf{y}$ . They yield precisely the same variational expression. However, both are useful because the first helps motivate a MAP prediction algorithm, while the second helps characterize our learning algorithm in Section 7 as (approximate) variational EM.

**Proposition 1.** *For fixed  $\boldsymbol{\theta}$  and  $L_\psi$ , the output  $\boldsymbol{\mu}^*$  of inference in the augmented objective (2) is equivalent to the output of standard inference (1) in an MRF with the same clique structure as our base model, but with a modified parameter  $\tilde{\boldsymbol{\theta}} = \boldsymbol{\theta} - \nabla L_\psi(\boldsymbol{\mu}^*)$ .*

*Proof.* Forming a Lagrangian for (2), the stationarity conditions with respect to the variable  $\boldsymbol{\mu}$  are:

$$0 = -(\boldsymbol{\theta} - \nabla L_\psi(\boldsymbol{\mu}^*)) - \nabla H_B(\boldsymbol{\mu}^*) + \nabla_\mu C(\boldsymbol{\mu}, \boldsymbol{\lambda}), \quad (6)$$

where  $C(\boldsymbol{\mu}, \boldsymbol{\lambda})$  are collected terms relating to the marginal polytope constraints. The proposition follows because (6) is the same as the stationarity conditions for

$$\boldsymbol{\mu}^* = \arg \min_{\boldsymbol{\mu} \in \mathcal{M}} -\langle \boldsymbol{\theta} - \nabla L_\psi(\boldsymbol{\mu}^*), \boldsymbol{\mu} \rangle - H_B(\boldsymbol{\mu}). \quad \square \quad (7)$$

Therefore, we can characterize a joint distribution over  $\mathbf{y}$  by first finding  $\boldsymbol{\mu}^*$  by solving (2) and then defining an MRF over  $\mathbf{y}$  with parameters  $\tilde{\boldsymbol{\theta}}$ . Even more conveniently, our inference technique in Section 6 iteratively estimates  $\tilde{\boldsymbol{\theta}}$  on the fly, namely via the dual iterate  $\boldsymbol{\theta}_t$  in Algorithm 1.

Ultimately, in many prediction problems we seek a single output configuration  $\mathbf{y}$  rather than an inferred distribution over outputs. Proposition 1 suggests a simple prediction procedure: first, find the variational distribution over  $\mathbf{y}$  parametrized as an MRF with parameter  $\tilde{\boldsymbol{\theta}}$ . Then, perform

MAP in this MRF. Assuming an available marginal inference routine for this MRF, we assume the tractability of MAP – for example using a dynamic program. We avoid predicting  $\mathbf{y}$  by locally maximizing nodes’ marginals, since this would not necessarily yield feasible outputs.

Instead of solving (2), we could have introduced global energy terms to the MAP objective (3) that act directly on values  $S(\mathbf{y})$  rather than on expectations  $\boldsymbol{\mu}$ , as in (2). However, this yields a difficult combinatorial optimization problem for prediction and does not yield a natural way to learn the parametrization of the global energy. Section 8.1 demonstrates that using energy terms defined on marginals, and performing MAP inference in the associated MRF, performs as well or better than an LP technique designed to directly perform MAP subject to global penalty terms.

Our second variational interpretation characterizes  $\boldsymbol{\mu}^*$  as a variational approximation to a complex joint distribution:

$$P_c(\mathbf{y}|\mathbf{x}) = (1/Z_{\boldsymbol{\theta},\psi})P_\theta(\mathbf{y}|\mathbf{x})P_\psi(\mathbf{y}|\mathbf{x}). \quad (8)$$

We assume that isolated marginal inference in  $P_\theta(\mathbf{y}|\mathbf{x})$  is tractable, while  $P_\psi(\mathbf{y}|\mathbf{x})$  is an alternative structured distribution over  $\mathbf{y}$  for which we do not have an efficient inference algorithm. Specifically, we assume that (1) can be solved for  $P_\theta$ . Furthermore, we assume that  $P_\psi(\mathbf{y}|\mathbf{x}) \propto \exp(L_\psi(S(\mathbf{y}); \mathbf{x}))$ , where  $L_\psi(\cdot; \mathbf{x})$  is a convex function, conditional on input features  $\mathbf{x}$ . Going forward, we will often suppress the dependence of  $L_\psi$  on  $\mathbf{x}$ . Above,  $Z_{\boldsymbol{\theta},\psi}$  is the normalizing constant of the combined distribution. Note that if  $L$  was linear, inference in both  $P_\psi(\mathbf{y}|\mathbf{x})$  and  $P_c(\mathbf{y}|\mathbf{x})$  would be tractable, since the distribution would decompose over the same cliques as  $P_\theta(\mathbf{y}|\mathbf{x})$ .

Not surprisingly, (8) is intractable to reason about, due to the non-local terms in (2), so we approximate it with a variational distribution  $Q(\mathbf{y})$ . The connection between this variational approximation and Proposition 1 is derived in Appendix A. Here, we assume no clique structure on  $Q(\mathbf{y})$ , but show that minimizing a variational upper bound on  $KL(Q(\mathbf{y})||P_c(\mathbf{y}|\mathbf{x}))$ , for a given  $\mathbf{x}$ , yields a  $Q$  that is parametrized compactly as the MRF in Proposition 1. We discuss the relationship between this and general mean-field inference in Section 5.

Although the analysis of this section assumes convexity of  $L_\psi$ , our inference techniques can be applied to non-convex  $L_\psi$ , as discussed in Section 6.3, and our learning algorithm produces state-of-the-art results even in the non-convex regime for a benchmark OCR task.

#### 5 RELATED MODELING TECHNIQUES

**Mean field** variational inference in undirected graphical models is a particular application of our inference framework, with a non-convex  $L_\psi$  (Wainwright & Jordan, 2008).



The technique estimates marginal properties of a complex joint distribution  $P$  using the clique marginals  $\mu$  of some tractable base distribution  $Q$ , not necessarily fully factorized. This induces a partitioning of the cliques of  $P$  into those represented directly by  $\mu$  and those where we define clique marginals as a product distribution of the relevant nodes' marginals in  $\mu$ . To account for the energy terms of the full model involving cliques absent in the simple base model, the energy  $\langle \theta, \mu \rangle$  of the base model is augmented with an extra function of  $\mu$ .

$$L(\mu) = - \sum_{c \in \mathcal{C}} \left\langle \theta_c, \bigotimes_{n \in c} \mu_n \right\rangle \quad (9)$$

where  $\mathcal{C}$  is the set of cliques not included in the tractable sub-model,  $\theta_c$  are the potentials of the original graphical model corresponding to the missing cliques, and  $\bigotimes_n \mu_n$  represents a repeated outer (tensor) product of the node marginals for the nodes in those cliques.

Note  $L(\mu)$  is non-linear and non-convex. Our work generalizes (9) by allowing arbitrary non-linear interaction terms between components of  $\mu$ . This is very powerful – for example, in our citation extraction experiments in Section 8.1, expressing these global terms in a standard graphical model would require many factors touching all variables. Local coordinate ascent mean-field can be frustrated by these rigid global terms. Our gradient-based method avoids these issues by updating all marginals simultaneously.

**Dual decomposition** is a popular method for performing MAP inference in complex structured prediction models by leveraging repeated calls to MAP in tractable submodels (Komodakis et al., 2007; Sontag et al., 2011). The family of models solvable with dual decomposition is limited, however, because the terms that link the submodels must be expressible as linear constraints. Similar MAP techniques (Ravikumar et al., 2010; Martins et al., 2011; Fu & Banerjee, 2013) based on the alternating direction method of multipliers (ADMM) can be adapted for marginal inference, in problems where marginal inference in submodels is tractable. However, the non-local terms are defined as linear functions on settings of graphical model nodes, while our non-linear  $L_\psi(\mu)$  terms provide practitioners with an expressive means to learn and enforce regularities of the inference output.

**Posterior regularization** (PR) (Ganchev et al., 2010), **learning from measurements** (LFM) Liang et al. (2009), and **generalized expectations** (GE) (Mann & McCallum, 2010), are a family of closely-related techniques for performing unsupervised or semi-supervised learning of a conditional distribution  $P_\theta(y|x)$  or a generative model  $P_\theta(x|y)$  using expectation-maximization (EM), where the E-step for latent variables  $y$  does not come directly from inference in the model, but instead from projection onto a set of expectations obeying global regularity properties. In PR

and GE, this yields a projection objective of the form (2), where the  $L_\psi$  terms come from a Lagrangian relaxation of regularity constraints, and  $\psi$  corresponds to dual variables. Originally, PR employed linear constraints on marginals, but He et al. (2013) extend the framework to arbitrary convex differentiable functions. Similarly, in LFM such an inference problem arises because we perform posterior inference assuming that the observations  $y$  have been corrupted under some noise model. Tarlow & Zemel (2012) also present a method for learning with certain forms of non-local losses in a max-margin framework.

Our goals are very different than the above learning methods. We do not impose non-local terms  $L_\psi$  in order to regularize our learning process or allow it to cope with minimal annotation. Instead, we use  $L_\psi$  to increase the expressivity of our model, performing inference for every test example, using a different  $\psi$ , since it depends on input features. Since we are effectively ‘learning the regularizer,’ on fully-labeled data, our learning approach in Section 7 differs from these methods. Finally, unlike these frameworks, we employ non-convex  $L_\psi$  terms in some of our experiments. The algorithmic consequences of non-convexity are discussed in Section 6.3.

## 6 OPTIMIZING THE NON-LOCAL MARGINAL INFERENCE OBJECTIVE

We now present an approach to solving (2) using non-Euclidean projected gradient methods, which require access to a procedure for marginal inference in the base distribution (which we term the *marginal oracle*), as well as access to the gradient of the energy function  $L_\psi$ . We pose these algorithms in the *composite minimization* framework, which gives us access to a wide variety of algorithms that are discussed in the supplementary material.

### 6.1 CONVEX OPTIMIZATION BACKGROUND

Before presenting our algorithms, we review several definitions from convex analysis (Rockafellar, 1997).

We call a function  $\varphi$   $\sigma$ -strongly convex with respect to a norm  $\|\cdot\|_P$ , if for all  $x, y \in \text{dom}(\varphi)$ ,

$$\varphi(y) \geq \varphi(x) + \nabla\varphi(x)^T(y-x) + \frac{\sigma}{2}\|y-x\|_P^2.$$

**Proposition 2** (e.g. Beck & Teboulle (2003)). *The negative entropy function  $-H(x) = \sum_i x_i \log x_i$  is 1-strongly convex with respect to the 1-norm  $\|\cdot\|_1$  over the interior of the simplex  $\Delta$  (restricting  $\text{dom}(H)$  to  $\text{int}(\Delta)$ ).*

Given a smooth and strongly convex function  $\varphi$ , we can also define an associated generalized (asymmetric) distance measure called the *Bregman divergence* (Bregman, 1967)

---

**Algorithm 1** Bethe-RDA

---

**Input:** parameters  $\theta$ , energy function  $L_{\psi}(\mu)$   
 set  $\theta_0 = \theta$   
 set  $\mu_0$  to prox-center MARGINAL-ORACLE( $\theta_0$ )  
 $\bar{g}_0 = 0$   
**repeat**  
 $\beta_t = \text{constant} \geq 0$   
 $\bar{g}_t = \frac{t-1}{t}\bar{g}_{t-1} + \frac{1}{t}\nabla L(\mu_t)$   
 $\theta_t = \theta - \frac{t}{t+\beta_t}\bar{g}_t$   
 $\mu_t = \text{MARGINAL-ORACLE}(\theta_t)$   
**until** CONVERGED( $\mu_t, \mu_{t-1}$ )

---

generated by  $\varphi$ ,

$$B_{\varphi}(x, x_0) = \varphi(x) - \varphi(x_0) - \langle \nabla \varphi(x_0), x - x_0 \rangle.$$

For example, the KL divergence is the Bregman divergence associated to the negative entropy function, and the squared Euclidean distance is its own associated divergence.

*Composite minimization* (Passty, 1979) is a family of techniques for minimizing functions of the form  $h = f + R$ , where we have an oracle that allows us to compute minimizations over  $R$  in closed form (usually  $R$  here takes the form of a regularizer). Problems of this form are often solved with an algorithm called *proximal gradient*, which minimizes  $h(x)$  over some convex set  $X$  using:

$$x_{t+1} = \arg \min_{x \in X} \langle \nabla f(x_t), x \rangle + \frac{1}{2\eta_t} \|x - x_t\|_2^2 + R(x),$$

for some decreasing sequence of learning rates  $\eta_t$ . Note that because of the requirement  $x \in X$ , proximal gradient generalizes projected gradient descent – since unconstrained minimization might take us out of the feasible region  $X$ , computing the update requires projecting onto  $X$ .

But there is no reason to use the squared Euclidean distance when computing our updates and performing the projection. In fact, the squared term can be replaced by any Bregman divergence. This family of algorithms includes the *mirror descent* and *dual averaging* algorithms (Beck & Teboulle, 2003; Nesterov, 2009).

We base our projected inference algorithms on *regularized dual averaging* (RDA) (Xiao, 2010). The updates are:

$$x_{t+1} = \arg \min_{x \in X} \langle \bar{g}_t, x \rangle + \frac{\beta_t}{t} \varphi(x) + R(x), \quad (10)$$

where  $\bar{g}_t = \frac{1}{t} \sum_k^t \nabla f(x_k)$  is the average gradient of  $f$  encountered so far. One benefit of RDA is that it does not require the use of a learning rate parameter ( $\beta_t = 0$ ) when using a strongly convex regularizer. RDA can be interpreted as doing a projection onto  $X$  using the Bregman divergence generated by the strongly convex function  $\varphi + R$ .

## 6.2 OUR ALGORITHM

These non-Euclidean proximal algorithms are especially helpful when we are unable to compute a projection in terms of Euclidean distance, but can do so using a different Bregman divergence. We will show that this is exactly the case for our problem of projected inference: the marginal oracle allows us to project in terms of KL divergence.

However, to maintain tractability we avoid using the entropy function  $H$  on the exponentially-large simplex  $\Delta$ , and instead optimize over the structured, factorized marginal polytope  $\mathcal{M}$  and its corresponding structured Bethe entropy  $H_{\mathcal{B}}$ . For tree-structured models,  $H$  and  $H_{\mathcal{B}}$  have identical values, but different inputs. It remains to show the strong convexity of  $-H_{\mathcal{B}}$  so we can use it in RDA.

**Proposition 3.** *For trees with  $n$  nodes, the negative Bethe entropy function  $-H_{\mathcal{B}}$  is  $\frac{1}{2}(2n-1)^{-2}$ -strongly convex with respect to the 2-norm over the interior of the marginal polytope  $\mathcal{M}$ .*

*Proof.* Consequence of Lemma 1 in Fu & Banerjee (2013).

With these definitions in hand, we present Bethe-RDA projected inference Algorithm 1. This algorithm corresponds to instantiating (10) with  $R = -H_{\mathcal{B}} - \langle \theta, \mu \rangle$  and  $\varphi = -H_{\mathcal{B}}$ . Note the simplicity of the algorithm when choosing  $\beta_t = 0$ . It is intuitively appealing that the algorithm amounts to no more than calling our marginal inference oracle with iteratively modified parameters.

**Proposition 4.** *For convex energy functions and convex  $-H_{\mathcal{B}}$ , the sequence of primal averages of Algorithm 1 converges to the optimum of the variational objective (2) with suboptimality of  $O(\frac{\ln(t)}{t})$  at time  $t$ .*

*Proof.* This follows from Theorem 3 of Xiao (2010) along with the strong convexity of  $-H_{\mathcal{B}}$ .  $\square$

If we have more structure in the energy functions, specifically a Lipschitz-continuous gradient, we can modify the algorithm to use Nesterov’s acceleration technique and achieve a convergence of  $O(\frac{1}{t^2})$ . Details can be found in Appendix D. Additionally, in practice these problems need not be solved to optimality and give stable results after a few iterations, as demonstrated in Figure 8.1.

## 6.3 INFERENCE WITH NON-CONVEX, NON-LOCAL ENERGIES

An analogy can be made here to loopy belief propagation – even in the case of non-convex loss functions (and even non-convex entropy functions with associated inexact marginal oracles), the updates of our inference (and learning) algorithms are well-defined. Importantly, since one of our motivations for developing non-local inference was to

---

**Algorithm 2** Learning with non-local energies

---

**Input:** examples  $\mathbf{x}_i, \mathbf{y}_i$  and inference oracle  $\text{MARG}()$  for distributions with the clique structure of  $P_\theta(\mathbf{y}|\mathbf{x})$ .

**Output:** parameters  $(\theta, \psi)$  for  $P_c(\mathbf{y}|\mathbf{x})$ .

**repeat**

//E-Step

**for all**  $(\mathbf{x}_i, \mathbf{y}_i)$  **do**

$\mu_i \leftarrow (\text{Algorithm 1})$  // using  $\theta, \psi$  and  $\text{MARG}()$

$\rho_i \leftarrow (\text{Proposition 5})$  // using  $\psi, \mu_i$

// note  $Q_i(\mathbf{y}_i)$  is a CRF with potentials  $\theta + \rho_i$ .

**end for**

//M-Step (gradient-based learning of CRF parameters)

**repeat**

$m_i \leftarrow \text{MARG}(Q_i) \forall j$  //standard CRF inference

$\nabla_\theta \leftarrow \sum_i S(\mathbf{y}_i) - m_i$

$\nabla_\psi \leftarrow \sum_i \frac{d\rho_i}{d\psi}^\top (S(\mathbf{y}_i) - m_i)$

$\theta \leftarrow \text{Gradient-Step}(\theta, \nabla_\theta)$

$\psi \leftarrow \text{Gradient-Step}(\psi, \nabla_\psi)$

**until** converged

**until** converged OR iter > max\_iters

---

---

**Algorithm 3** Doubly-stochastic learning with  $L_\psi$  given by a sum of scalar functions of linear measurements (5).

---

**Input:** examples  $\mathbf{x}_i, \mathbf{y}_i$  and  $\text{MARGINAL-ORACLE}()$  for distributions with the clique structure of  $P_\theta(\mathbf{y}|\mathbf{x})$ .

**Output:** parameters  $(\theta, \psi)$  for  $P_c(\mathbf{y}|\mathbf{x})$ .

**repeat**

sample  $(\mathbf{x}_i, \mathbf{y}_i)$  randomly

$\mu_i \leftarrow (\text{Algorithm 1})$

$\nabla_\theta \leftarrow S(\mathbf{y}_i) - \mu_i$

$\nabla_{\psi_j} \leftarrow \nabla_{\ell_j}(\mu_i) a_j^\top (S(\mathbf{y}_i) - \mu_i)$

$\theta \leftarrow \text{Gradient-Step}(\theta, \nabla_\theta)$

$\psi \leftarrow \text{Gradient-Step}(\psi, \nabla_\psi)$

**until** converged OR iter > max\_iters

---

generalize mean field inference, and the additional penalty terms are non-convex in that case, we would like our algorithms to work for the non-convex case as well.

Unlike loopy belief propagation, however, since we derive our algorithms in the framework of composition minimization, we have access to a wealth of theoretical guarantees. Based on results from the theory of optimization with first-order surrogate loss functions (Mairal, 2013), in Appendix C we propose a small modification to Algorithm 1 with an asymptotic convergence condition even for non-convex energies. In practice we find that the unmodified Algorithm 1 also works well for these problems, and experimentally in Section 8.2, we see good performance in both inference and learning with non-convex energy functions.

## 7 LEARNING MODELS WITH NON-LOCAL ENERGIES

We seek to learn the parameters  $\theta$  and  $\psi$  of the underlying CRF base model and  $L_\psi$ , respectively. Let  $S = \{\mathbf{y}_i, \mathbf{x}_i\}$  be  $n$  training examples. Let  $Q(\mathbf{y}_i; \mu_i)$  be the variational distribution for  $\mathbf{y}_i$  resulting from applying Proposition 1. Namely,  $Q(\mathbf{y}_i; \mu_i)$  is an MRF with parameters

$$\rho_i := \theta - \nabla_{\mu} L_\psi(\mu_i). \quad (11)$$

We employ the notation  $Q(\mathbf{y}_i; \mu_i)$  to highlight the role of  $\mu_i$ : for a given  $(\mathbf{y}_i, \mathbf{x}_i)$  pair, the family of variational distributions over  $\mathbf{y}_i$  is indexed by possible values of  $\mu_i$  (recall we suppress the explicit dependence of  $\theta$  and  $\psi$  on  $\mathbf{x}$ ). Finally, define the shorthand  $M = \{\mu_1, \dots, \mu_n\}$ .

$\psi$  interacts with the data in a complex manner that prevents us from using standard learning techniques for the exponential family. Namely, we can not easily differentiate a likelihood with respect to  $\psi$ , since this requires differentiating the output  $\mu$  of a convex optimization procedure, and the extra  $L_\psi$  term in (2) prevents the use of conjugate duality relationships available for the exponential family. We could have used automatic methods to differentiate the iterative inference procedure (Stoyanov et al., 2011; Domke, 2012), but found our learning algorithm works well.

We employ a variational learning algorithm, presented in Algorithm 2, alternately updating the parameters  $M$  of our tractable CRF-structured variational distributions, and updating the parameters  $(\theta, \psi)$  assuming the following surrogate likelihood given by these CRF approximations:

$$L(\theta, \psi; M) = \sum_i \log Q(\mathbf{y}_i; \mu_i). \quad (12)$$

Given  $\theta$  and  $\psi$ , we update  $M$  using Algorithm 1. Given  $M$ , we update  $\theta$  and  $\psi$  by taking a single step in the direction of the gradient of the surrogate likelihood (12). We avoid taking more than one gradient step, since the gradients for  $\theta$  and  $\psi$  depend on  $M$  and an update to  $\theta$  and  $\psi$  will break the property that  $\mu(Q(\mathbf{y}; \mu_i)) = \mu_i$ . Therefore, we recompute  $\mu_i$  every time we update the parameters.

Overall, it remains to show how to compute gradients of (12). For  $\theta$ , we have the standard CRF likelihood gradient (Sutton & McCallum, 2006):

$$\nabla_\theta L(\theta, \psi; M) = \sum_i S(\mathbf{y}_i) - \mu_i. \quad (13)$$

For  $\psi$ , we have:

$$\nabla_\psi L(\theta, \psi; M) = \sum_i \frac{d\rho_i}{d\psi} \frac{d}{d\rho_i} \log Q(\mathbf{y}_i; \mu_i). \quad (14)$$

From (11),  $\frac{d}{d\rho_i} \log Q(\mathbf{y}_i; \mu_i)$  is also  $S(\mathbf{y}_i) - \mu_i$  and

$$\frac{d\rho_i}{d\psi} = \frac{d}{d\psi} \frac{d}{d\mu} L_\psi(\mu) \quad (15)$$

Clearly, this depends on the structure of  $L_\psi$ . Consider the parametrization (4). With this, we have:

$$\frac{\partial}{\partial \psi_j} \frac{d}{d\boldsymbol{\mu}} L_\psi(\boldsymbol{\mu}) = \nabla \ell_j(\boldsymbol{\mu}) \frac{d}{d\boldsymbol{\mu}} \ell_j(\boldsymbol{\mu}) \quad (16)$$

Therefore, we have  $\frac{\partial}{\partial \psi_j} \log Q(\mathbf{y}_i; \boldsymbol{\mu}_i) = \nabla \ell_j(\boldsymbol{\mu}) \frac{d}{d\boldsymbol{\mu}} \ell_j(\boldsymbol{\mu})^\top (S(\mathbf{y}) - \boldsymbol{\mu}_i)$ . For linear measurements (5), this amounts to

$$\nabla \ell(\boldsymbol{\mu}) (a_j^\top S(\mathbf{y}) - a_j^\top \boldsymbol{\mu}_i). \quad (17)$$

This has a simple interpretation: the gradient with respect to  $\psi_j$  equals the gradient of the scalar loss  $\ell_j$  at the current marginals  $\boldsymbol{\mu}_j$  times the difference in linear measurements between the ground truth labels and the inferred marginals.

Algorithm 2 has an expensive double-loop structure. In practice it is sufficient to employ a ‘doubly-stochastic’ version given in Algorithm 3, where we sample a training example  $(\mathbf{x}_i, \mathbf{y}_i)$  and use this to only perform a single gradient step on  $\boldsymbol{\theta}$  and  $\psi$ . To demonstrate the simplicity of implementing our learning algorithm, we avoid any abstract derivative notation in Algorithm 3 by specializing it to the case of (17). In our experiments, however, we sometimes do not use linear measurements. Overall, all our experiments use the fast doubly-stochastic approach of Algorithm 3 solely, since it performs well. In general, our learning algorithms are not guaranteed to converge because we approximate the complex interaction between  $\psi$  and  $\boldsymbol{\mu}$  with alternating updates. In practice, however, terminating after a fixed number of iterations yields models that generalize well.

Finally, recall that the notation  $L_\psi(\boldsymbol{\mu}_i)$  suppresses the potential dependence of  $\psi$  on  $\mathbf{x}_i$ . We assume each  $\psi_j$  is a differentiable function of features of  $\mathbf{x}_i$ . Therefore, in our experiments where  $\psi$  depends on  $\mathbf{x}_i$ , we perform gradient updates for the parametrization of  $\psi(\mathbf{x})$  via further application of the chain rule.

## 8 EXPERIMENTS

### 8.1 CITATION EXTRACTION

| Model                            | F1    |
|----------------------------------|-------|
| Our Baseline                     | 94.47 |
| Non-local Energies               | 95.47 |
| Baseline (Anzaroot et al., 2014) | 94.41 |
| Soft-DD (Anzaroot et al., 2014)  | 95.39 |

Table 1: Comparison of F1 scores on Citation Extraction dataset. We compare MAP inference F1 scores of our non-local energy model and the specialized dual decomposition model of Anzaroot et al. (2014). Both variants learn global regularities that significantly improve performance.

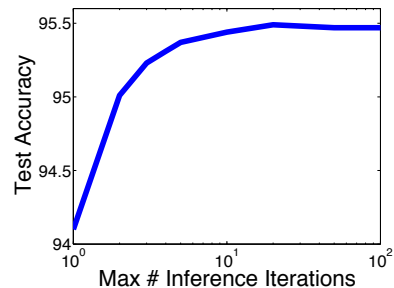


Figure 1: Citation extraction F1 when limiting maximum number of test-time inference iterations. Most of our accuracy gain is captured within the first 5-10 iterations.

We first apply our algorithm to the NLP task of performing text field segmentation on the UMass citation dataset (Anzaroot & McCallum, 2013), which contains strings of citations from research papers, segmented into fields (author, title, etc.). Our modeling approach, closely follows Anzaroot et al. (2014), who extract segmentations using a linear-chain segmentation model, to which they add a large set of ‘soft’ linear global regularity constraints.

Let  $\mathbf{y}$  be a candidate labeling. Imagine, for example, that we constrain predicted segmentations to have no more predicted last names than first names. Then, the numbers of first and last names can be computed by linear measurements  $a_{\text{first}}^\top S(\mathbf{y})$  and  $a_{\text{last}}^\top S(\mathbf{y})$ , respectively. A hard constraint on  $\mathbf{y}$  would enforce  $a_{\text{first}}^\top S(\mathbf{y}) - a_{\text{last}}^\top S(\mathbf{y}) = 0$ . This is relaxed in Anzaroot et al. (2014) to a penalty term

$$c \ell_h (a_{\text{first}}^\top S(\mathbf{y}) - a_{\text{last}}^\top S(\mathbf{y})) \quad (18)$$

that is added to the MAP inference objective, where  $\ell_h(x) = \max(1 - x, 0)$  is a hinge function. For multiple soft constraints, the overall prediction problem is

$$\arg \min_{\mathbf{y}} \langle -\boldsymbol{\theta}, S(\mathbf{y}) \rangle + \sum_j c_j \ell_h (a_j^\top S(\mathbf{y})), \quad (19)$$

where  $\boldsymbol{\theta}$  are the parameters of the underlying linear-chain model. They use a dual decomposition style algorithm for solving (19), that crucially relies on the specific structure of the hinge terms  $\ell_h$ . They learn the  $c_j$  for hundreds of ‘soft constraints’ using a perceptron-style algorithm.

We consider the same set of measurement vectors  $a_j$ , but impose non-local terms that act on *marginals*  $\boldsymbol{\mu}$  rather than specific values  $\mathbf{y}$ . Further, we use *smoothed* hinge functions, which improve the convergence rate of inference (Rennie, 2005). We find the variational distribution by solving the marginal inference version of (19), an instance of our inference framework with linear measurements (5):

$$\arg \min_{\boldsymbol{\mu}} \langle -\boldsymbol{\theta}, \boldsymbol{\mu} \rangle - H_B(\boldsymbol{\mu}) + \sum_j c_j \ell_h (a_j^\top \boldsymbol{\mu}), \quad (20)$$

As in Anzaroot et al. (2014), we first learn chain CRF parameters  $\theta$  on the training set. Then, we learn the  $c_j$  parameters on the development set, using Algorithm 3, and tune hyperparameters for development set performance. At both train and test time, we ignore any terms in (20) for which  $c_j < 0$ .

We present our results in Table 1, measuring segment-level F1. We can see that our baseline chain has slightly higher accuracy than the baseline approach of Anzaroot et al. (2014), possibly due to optimization differences. Our augmented model (Non-Local Energies) matches and very slightly beats their soft dual decomposition (Soft-DD) procedure. This is especially impressive because they employ a specialized linear-programming solver and learning algorithm adapted to the task of MAP inference under hinge-loss soft constraints, whereas we simply plug in our general learning and inference algorithms for non-local structured prediction – applicable to any set of energy functions.

Our comparable performance provides experimental evidence for our intuition that preferences about MAP configurations can be expressed (and “relaxed”) as functions of expectations. Anzaroot et al. (2014) solve a penalized MAP problem directly, while our prediction algorithm first finds a distribution satisfying these preferences, and then performs standard MAP inference in that distribution.

Finally, in Figure 1 we present results demonstrating that our algorithm’s high performance can be obtained using only 5-10 calls per test example to inference in the underlying chain model. In Section B, we analyze the empirical convergence behavior of Algorithm 1.

## 8.2 HANDWRITING RECOGNITION

| N-Grams  | 2     | 3     | 4     | 5     | 6     |
|----------|-------|-------|-------|-------|-------|
| Accuracy | 85.02 | 96.20 | 97.21 | 98.27 | 98.54 |

Table 2: Character-wise accuracy of Structured Prediction Cascades (Weiss et al., 2012) on OCR dataset.

| Model                    | Accuracy     |
|--------------------------|--------------|
| 2-gram (base model)      | 84.93        |
| $L_{\psi}^u$             | 94.01        |
| $L_{\psi}^u$ (MM)        | 94.96        |
| $L_{\psi}^w$             | 98.26        |
| $L_{\psi}^w$ (MM)        | <b>98.83</b> |
| 55-Class Classifier (MM) | 86.06        |

Table 3: Character-wise accuracy of our baselines, and models using learned non-local energies on Handwriting Recognition dataset. Note that word classifier baseline is also given in character-wise accuracy for comparison.

We next apply our algorithms to the widely-used handwrit-

ing recognition dataset of Taskar et al. (2004). We follow the setup of Weiss et al. (2012), splitting the data into 10 equally sized folds, using 9 for training and one to test. We report the cross-validation results across all 10 folds.

The *structured prediction cascades* of Weiss et al. (2012) achieve high performance on this dataset by using extremely high order cliques of characters (up to 6-grams), for which they consider only a small number of candidate outputs. Their state-of-the-art results are reproduced in Table 2. The excellent performance of these large-clique models is consequence of the fact that the data contains only 55 unique words, written by 150 different people. Once the model has access to enough higher-order context, the problem becomes much easier to solve.

With this in mind, we design two non-convex, non-local energy functions. These energies are intended to regularize our predictions to lie close to known elements of the vocabulary. Our base model is a standard linear-chain CRF with image features on the nodes, and no features on the bigram edge potentials. Let  $U(\mu) = \sum_n \mu_n$  be a function that takes the concatenated vector of node and edge marginals and sums up all of the node marginals, giving the global unigram expected sufficient statistics. Let  $\{u_i\} = \{U(\mu(y_i))\}$  indicate the set of all such unique vectors when applying  $U$  to the train set empirical sufficient statistics for each data case  $y_i$ . Simply, this gives 55 vectors  $u_i$  of length 26 containing the unigram counts for each unique word in the train set.

Our intuition is that we would like to be able to “nudge” the results of inference in our chain model by pulling the inferred  $U(\mu)$  to be close to one of these global statistics vectors. We add the following non-convex non-local energy function to the model:

$$L_{\psi}^u(\mu) = \psi \min_i \|u_i - U(\mu)\|_1. \quad (21)$$

We learn two variants of this model, which differently parametrize the dependence of  $\psi$  on  $x$ . The first has a single bias feature on the non-local energy. The second conditions on a global representation of the sequence: concretely, we approximate the RBF *kernel mean map* (MM) (Smola et al., 2007) using random Fourier features (RFF) (Rahimi & Recht, 2007). This simply involves multiplying each image feature vector in the sequence by a random matrix with  $\sim 1000$  rows, applying a pointwise non-linearity, and taking  $\psi$  to be a linear function of the average vector.

Results of these experiments can be seen in Table 3. Adding the non-local energy brings our performance well above the baseline bigram chain model, and our training procedure is able to give substantially better performance when  $\psi$  depends on the above input features.

The energy  $L_{\psi}^u$ , based on unigram sufficient statistics, is not able to capture the relative ordering of letters in the vocabulary words, which the structured prediction cascades

| $s$        | 625  | 10k | 50k |
|------------|------|-----|-----|
| Our Method | 0.19 | 2.7 | 14  |
| IP         | 2.8  | 93  | 690 |

Table 4: Comparison of runtime (in seconds, averaged over 10 trials) between the interior point solver (IP) of Sheldon et al. (2013) v.s. Algorithm 1 on different CGM problem sizes  $s$ , the cardinality of the edge potentials in the underlying graphical model, where marginal inference is  $O(s)$ .

models do capture. This motivates us to consider another energy function. Let  $\{w_i\} = \{\mu_n(y_i)\}$  be the set of unique vectors of concatenated node marginal statistics for the train set. This gives 55 vectors of length  $l_i * 26$ , where  $l_i$  is the length of the  $i$ th distinct train word. Next, we define a different energy function to add to our base chain model:

$$L_{\psi}^w(\mu) = \psi \min_i \|w_i - \mu\|_1. \quad (22)$$

Once again we implement featurized and non-featurized versions of this model. As noted in structured prediction cascades, giving the model access to this level of high-order structure in the data makes the inference problem extremely easy. Our model outperforms the best structured prediction cascades results, and we note again an improvement from using the featurized over the non-featurized  $\psi$ .

Of course, since the dataset has only 55 actual labels, and some of those are not valid for different input sequences due to length mismatches, this is arguably a classification problem as much as a structured prediction problem. To address this, we create another baseline, which is a constrained 55-class logistic regression classifier (constrained to only allow choosing output classes with appropriate lengths given the input). We use our same global mean-map features from the  $L_{\psi}^*$  ( $MM$ ) variants of the structured model and report these results in Table 3. We also tune the number of random Fourier features as a hyperparameter to give the classifier as much expressive power as possible. As we can see, the performance is still significantly below the best structured models, indicating that the interplay between local and global structure is important.

### 8.3 COLLECTIVE GRAPHICAL MODELS

Next, we demonstrate that that our proximal gradient-based inference framework dramatically speeds up approximate inference in *collective graphical models* (CGMs) (Sheldon & Dietterich, 2011). CGMs are a method for structured learning and inference with noisy aggregate observation data. The large-scale dependency structure is represented via a graphical model, but the nodes represent not just single variables, but aggregate sufficient statistics of large sets of underlying variables, corrupted by some noise model. In previous work, CGMs have been successfully applied to modeling bird migration. Here, the base model is a lin-

ear chain representing a time series of bird locations. Each observed variable corresponds to counts from bird watchers in different locations. These observations are assumed to be Poisson distributed with rate proportional to the true count of birds present. The CGM MAP task is to infer the underlying migration patterns.

Sheldon et al. (2013) demonstrate that MAP in CGMs is NP-hard, *even for trees*, but that approximate MAP can be performed by solving a problem of the form (2):

$$\mu^* = \arg \max_{\mu} \langle \theta, \mu \rangle + H_B(\mu) + \sum_i^n P_i(\mu_i | \psi y_i) \quad (23)$$

where  $P_i$  are (concave) Poisson log-likelihoods and each  $y_i$  is an observed bird count.

For the case where the underlying CGM graph is a tree, the ‘hard EM’ learning algorithm of Sheldon et al. (2013) is the same as Algorithm 2 specialized to their model. Therefore, Sheldon et al. (2013) provide additional experimental evidence that our alternating surrogate-likelihood optimization works well in practice.

The learning procedure of Sheldon et al. (2013) is very computationally expensive because they solve instances of (23) using an interior-point solver in the inner loop. For the special case of trees, Algorithm 1 is directly applicable to (23). Using synthetic data and code obtained from the authors, we compare their generic solver to Algorithm 1 for solving instances of (23). In Table 4, we see that our method achieves a large speed-up with no loss in solution accuracy (since it solves the same convex problem).

## 9 DISCUSSION AND FUTURE WORK

Our results show that our inference and learning framework allows for tractable modeling of non-local dependency structures, resistant to traditional probabilistic formulations. By approaching structured modeling not via independence assumptions, but as arbitrary penalty functions on the marginal vectors  $\mu$ , we open many new modeling possibilities. Additionally, our generic gradient-based inference method can achieve substantial speedups on pre-existing problems of interest. In future work, we will apply our framework to new problems and new domains.

### ACKNOWLEDGEMENTS

This work was supported in part by the Center for Intelligent Information Retrieval, in part by DARPA under agreement number FA8750-13-2-0020, and in part by NSF grant #CNS-0958392. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

## References

- Anzaroot, Sam and McCallum, Andrew. A new dataset for fine-grained citation field extraction. In *ICML Workshop on Peer Reviewing and Publishing Models*, 2013.
- Anzaroot, Sam, Passos, Alexandre, Belanger, David, and McCallum, Andrew. Learning soft linear constraints with application to citation field extraction. In *ACL*, 2014.
- Beck, Amir and Teboulle, Marc. Mirror descent and non-linear projected subgradient methods for convex optimization. *Operations Research Letters*, 31(3):167–175, 2003.
- Bregman, Lev M. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR computational mathematics and mathematical physics*, 7(3):200–217, 1967.
- Domke, Justin. Generic methods for optimization-based modeling. In *AISTATS*, 2012.
- Duchi, John, Shalev-Shwartz, Shai, Singer, Yoram, and Tewari, Ambuj. Composite objective mirror descent. In *COLT*, 2010.
- Fu, Qiang and Banerjee, Huahua Wang Arindam. Bethadmm for tree decomposition based parallel map inference. In *UAI*, 2013.
- Ganchev, Kuzman, Graça, Joao, Gillenwater, Jennifer, and Taskar, Ben. Posterior regularization for structured latent variable models. *JMLR*, 99:2001–2049, 2010.
- He, L., Gillenwater, J., and Taskar, B. Graph-Based Posterior Regularization for Semi-Supervised Structured Prediction. In *CoNLL*, 2013.
- Komodakis, Nikos, Paragios, Nikos, and Tziritas, Georgios. Mrf optimization via dual decomposition: Message-passing revisited. In *IEEE ICCV*, 2007.
- Lafferty, John, McCallum, Andrew, and Pereira, Fernando CN. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001.
- Liang, Percy, Jordan, Michael I, and Klein, Dan. Learning from measurements in exponential families. In *ICML*, 2009.
- Mairal, Julien. Optimization with first-order surrogate functions. In *ICML*, 2013.
- Mann, Gideon S and McCallum, Andrew. Generalized expectation criteria for semi-supervised learning with weakly labeled data. *JMLR*, 11:955–984, 2010.
- Martins, André, Figueiredo, Mário, Aguiar, Pedro, Smith, Noah A, and Xing, Eric P. An augmented lagrangian approach to constrained map inference. In *ICML*, 2011.
- Nesterov, Yurii. Primal-dual subgradient methods for convex problems. *Mathematical programming*, 120(1):221–259, 2009.
- Passty, Gregory B. Ergodic convergence to a zero of the sum of monotone operators in hilbert space. *Journal of Mathematical Analysis and Applications*, 72(2):383–390, 1979.
- Rahimi, Ali and Recht, Benjamin. Random features for large-scale kernel machines. In *NIPS*, 2007.
- Ravikumar, Pradeep, Agarwal, Alekh, and Wainwright, Martin J. Message-passing for graph-structured linear programs: Proximal methods and rounding schemes. *JMLR*, 11:1043–1080, 2010.
- Rennie, Jason DM. Smooth hinge classification, 2005.
- Rockafellar, R Tyrell. *Convex Analysis*, volume 28. Princeton University Press, 1997.
- Sheldon, Daniel, Sun, Tao, Kumar, Akshat, and Dietterich, Thomas G. Approximate inference in collective graphical models. In *ICML*, 2013.
- Sheldon, Daniel R and Dietterich, Thomas G. Collective graphical models. In *NIPS*, 2011.
- Smola, Alex, Gretton, Arthur, Song, Le, and Schölkopf, Bernhard. A hilbert space embedding for distributions. In *Algorithmic Learning Theory*, pp. 13–31. Springer, 2007.
- Sontag, David, Globerson, Amir, and Jaakkola, Tommi. Introduction to dual decomposition for inference. *Optimization for Machine Learning*, 1:219–254, 2011.
- Stoyanov, Veselin, Ropson, Alexander, and Eisner, Jason. Empirical risk minimization of graphical model parameters given approximate inference, decoding, and model structure. In *AISTATS*, 2011.
- Sutton, Charles and McCallum, Andrew. An introduction to conditional random fields for relational learning. *Introduction to statistical relational learning*, pp. 93–128, 2006.
- Tarlow, Daniel and Zemel, Richard S. Structured output learning with high order loss functions. In *AISTATS*, 2012.
- Taskar, Ben, Carlos, Guestrin, and Koller, Daphne. Max-margin markov networks. In *NIPS*, 2004.
- Wainwright, Martin J and Jordan, Michael I. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, (1-2):1–305, 2008.
- Weiss, D., Sapp, B., and Taskar, B. Structured Prediction Cascades. *ArXiv e-prints*, August 2012.
- Xiao, Lin. Dual averaging methods for regularized stochastic learning and online optimization. *JMLR*, 11:2543–2596, 2010.

---

# A Smart-Dumb/Dumb-Smart Algorithm for Efficient Split-Merge MCMC

---

**Wei Wang**  
LIP6

Université Pierre et Marie Curie, 75005 Paris  
benwei.wang@outlook.com

**Stuart Russell**

Computer Science Division  
University of California, Berkeley, CA 94720  
russell@cs.berkeley.edu

## Abstract

Split-merge moves are a standard component of MCMC algorithms for tasks such as multi-target tracking and fitting mixture models with unknown numbers of components. Achieving rapid mixing for split-merge MCMC has been notoriously difficult, and state-of-the-art methods do not scale well. We explore the reasons for this and propose a new split-merge kernel consisting of two sub-kernels: one combines a “smart” split move that proposes plausible splits of heterogeneous clusters with a “dumb” merge move that proposes merging random pairs of clusters; the other combines a dumb split move with a smart merge move. We show that the resulting smart-dumb/dumb-smart (SDDS) algorithm outperforms previous methods. Experiments with entity-mention models and Dirichlet process mixture models demonstrate much faster convergence and better scaling to large data sets.

## 1 INTRODUCTION

Markov Chain Monte Carlo (MCMC) algorithms have become a central pillar of statistical inference and machine learning. MCMC algorithms repeatedly apply a stochastic *kernel* transformation to an initial state, generating a random walk through the sample space whose stationary distribution matches a desired target distribution. Within the general Metropolis-Hastings (MH) family of MCMC methods, which includes many specific algorithms that have proven useful in practice, the kernel is built from a *proposal* step followed by a stochastic *acceptance* step whose probability is determined by the chosen proposal distribution. One key to efficient MH inference, then, is proposal design.<sup>1</sup>

---

<sup>1</sup>Other approaches include parallelization [Chang and Fisher, 2013; Williamson *et al.*, 2013] and taking advantage of symme-

This paper focuses on MH proposal design for *split-merge* moves. Split and merge moves, which form a complementary pair comprising a kernel, are useful for problems where an MCMC state can be thought of as consisting of a number of components or clusters, each of which is responsible for some subset of observations. The canonical example is the family of *mixture models*: a split move in such a model converts one mixture component into two, dividing the observations of the original component between the two new components; a merge move combines two components and their observations into a single component [Dahl, 2003; Pasula *et al.*, 2003; Jain and Neal, 2004]. Split-merge moves are also common in multitarget tracking, where a “component” is a single track joining together observations of an object over multiple time steps [Pasula *et al.*, 1999; Khan *et al.*, 2005]. The state of the art for split-merge MCMC in mixture models is considered to be the *restricted Gibbs split-merge* (RGSM) algorithm of Jain and Neal [2004].

The general idea followed in most MH proposal designs is to make the proposal “smart” by preferentially proposing states with higher probability according to the target distribution. This tends to give higher acceptance probabilities. The Gibbs sampler [Geman and Geman, 1984] is the quintessential smart proposal: by proposing values for some subset of variables exactly in proportion to their probability, Gibbs sampling has an acceptance probability of 1. In the context of split-merge moves, a smart split would be one that favors splitting a heterogeneous cluster to produce two more homogeneous ones, and a smart merge would favor merging similar clusters to ensure a homogeneous result. As our analysis and experiments show, however, combining smart split and merge moves does not lead to high acceptance probabilities and rapid convergence. The reason is the asymmetry between subspaces with  $K$  and  $K + 1$  components: there are far more states in the latter than the former, whereas in the case of Gibbs sampling the source and target subspaces are identical.

---

try [Niepert and Domingos, 2014]. The work reported in this paper can easily be combined with these approaches.



Our proposed solution, the smart-dumb/dumb-smart (SDDS) algorithm, combines two kernels in parallel: one has a smart split move and a “dumb” merge move that proposes merging random pairs of clusters; the other combines a dumb split move with a smart merge move. We show that the resulting algorithm performs well in practice, maintaining high acceptance rates and converging reasonably quickly even for large data sets with many clusters, where RGSM and other algorithms fail. To our knowledge, the closest relative of SDDS is a parallel-computation MCMC algorithm due to Chang and Fisher [2013], who mention the use of a random split move as the complement of a merge move in one part of a rather complex algorithm. A second contribution of our paper is a fast, exact method for sampling a split move from the posterior over all possible splits of a given component, i.e., an efficient block-Gibbs proposal for splits.

The paper begins (Section 2) with background material on MCMC. Section 3 describes, for expository purposes, the *entity/mention model* (EMM), a very simple Bayesian mixture model with observations that are discrete tokens, and examines split-merge MCMC in the context of the EMM. Section 4 describes the SDDS algorithm in detail. Finally, Section 5 evaluates SDDS in comparison to other approaches, both on EMM data and on data from a Dirichlet process mixture model (DPMM).

## 2 MCMC METHODS

Here we provide a brief review of the relevant aspects of Metropolis–Hastings MCMC. Let  $\mathcal{X}$  be a sample space (a set of possible worlds); a sample point  $\mathbf{x} \in \mathcal{X}$  will be called a *state*. Let  $\pi(\cdot)$  be a *target distribution* of interest, such as the posterior distribution on  $\mathcal{X}$  given some evidence. The goal is to generate samples from  $\pi$ , or something close to it, so as to answer queries. A standard MCMC algorithm constructs a Markov chain from a *transition kernel*  $P(\mathbf{x}'|\mathbf{x})$ , such that the unique stationary distribution of the chain is  $\pi$ ; of primary concern is the rate of convergence of the Markov chain to its stationary distribution.

The Metropolis–Hastings (MH) algorithm [Metropolis *et al.*, 1953; Hastings, 1970] is a general template for building transition kernels with the desired property. Each transition is built from two steps: first, a new state  $\mathbf{x}'$  is proposed from a *proposal distribution*  $q(\mathbf{x}'|\mathbf{x})$ , then the new state is accepted with a probability given by

$$\alpha(\mathbf{x}'|\mathbf{x}) = \min\left\{1, \frac{\pi(\mathbf{x}')}{\pi(\mathbf{x})} \frac{q(\mathbf{x}|\mathbf{x}')}{q(\mathbf{x}'|\mathbf{x})}\right\}. \quad (1)$$

(If the proposal is not accepted, the new state is the same as the current state.) The ratio appearing in this expression is called the *MH ratio*; we have written it as the product of the *state ratio*  $\pi(\mathbf{x}')/\pi(\mathbf{x})$  and the *proposal ratio*  $q(\mathbf{x}|\mathbf{x}')/q(\mathbf{x}'|\mathbf{x})$ . The only “free parameter” in designing

an MH algorithm is the proposal  $q(\cdot|\cdot)$ , and it is this that determines the rate of convergence.

Gibbs sampling can be understood as a special case of MH [Gelman, 1992]. In its simplest form, it chooses a variable  $X_i$  uniformly at random from the set of  $n$  variables whose values define the state  $\mathbf{x}$  and proposes a value  $x'_i$  from the distribution  $\pi(X_i|\mathbf{x}_{-i})$ , where  $\mathbf{x}_{-i}$  denotes the current values for all variables other than  $X_i$ . Because this proposal distribution is proportional to the state probability, the proposal ratio for Gibbs is exactly the inverse of the state ratio. For the case where  $x_i$  and  $x'_i$  coincide, both ratios are 1; when they differ, we have

$$\begin{aligned} \frac{q(\mathbf{x}|\mathbf{x}')}{q(\mathbf{x}'|\mathbf{x})} &= \frac{q(x_i, \mathbf{x}_{-i}|x'_i, \mathbf{x}_{-i})}{q(x'_i, \mathbf{x}_{-i}|x_i, \mathbf{x}_{-i})} = \frac{\frac{1}{n}\pi(x_i|\mathbf{x}_{-i})}{\frac{1}{n}\pi(x'_i|\mathbf{x}_{-i})} \\ &= \frac{\pi(x_i, \mathbf{x}_{-i})\pi(\mathbf{x}_{-i})}{\pi(x'_i, \mathbf{x}_{-i})\pi(\mathbf{x}_{-i})} = \frac{\pi(\mathbf{x})}{\pi(\mathbf{x}')}. \end{aligned} \quad (2)$$

Hence the acceptance probability in (1) is therefore exactly 1 for Gibbs sampling.

Having a high acceptance probability—or at least, one bounded away from zero—is a necessary but not sufficient condition for rapid mixing in MH. Moves can be accepted with high probability but if those moves fail to lead the chain from one local maximum in  $\pi$  to another, overall mixing may still be slow. Thus, good proposal design is concerned with both acceptance probabilities and the ability to traverse the state space without getting stuck in local maxima.

## 3 THE ENTITY/MENTION MODEL

The entity/mention model or EMM is a very simple form of mixture model, defined here for the purposes of exposition. The EMM posits a certain (unknown) number of entities that are referred to by some set of mentions. For example, there is a person who may variously be referred to as “Barack Obama”, “the President”, “POTUS”, and so on. Given a collection of mentions of various entities—for example, in newspaper text—the task is to figure out how many entities exist, which mentions refer to which entities, and thence the ways in which any given entity may be mentioned. In its simplest form, the EMM assumes that each mention is a token with no internal structure, drawn from a fixed, known set of tokens. This renders the model less interesting than the models used in NLP research, but has the advantage of simplifying the analysis.

### 3.1 The EMM probability model

We assume  $N$  mentions and  $L$  possible tokens. An EMM model is composed from the following variables and conditional distributions:

- $K$ , the number of entities, drawn from a prior  $P(K)$ .

- For each entity  $k$ , a *dictionary*  $\theta_k$ , i.e., a categorical distribution over  $L$  tokens, drawn from a *Dirichlet*( $\alpha_k$ ). The set of dictionaries  $\{\theta_1, \dots, \theta_K\}$  is represented by  $\Theta$ .
- For each mention  $m_n$ , the entity  $S_n$  for that mention is drawn u.a.r. from the set of  $K$  entities, and the token for that entity is drawn from  $\theta_{S_n}$ . The (unknown) entities  $\{S_1, \dots, S_N\}$  are represented by  $\mathbf{S}$  and the observed mentions  $\{m_1, \dots, m_N\}$  by  $\mathbf{m}$ .

The EMM resembles a topic model for a single document with an unknown number of topics (entities).

In the experiments described below, we use a broad prior for  $K$ , namely a discretized log-normal distribution with the location parameter  $\mu$  and the scale parameter  $\sigma$  on a logarithmic scale:

$$P(K) = \frac{1}{C} \frac{1}{K\sigma\sqrt{2\pi}} e^{-(\log K - \mu)^2 / 2\sigma^2}$$

where  $C$  is an additional normalization factor arising from discretizing the distribution.

Because the entity for any given mention is assumed to be chosen u.a.r. from the available entities, we have

$$P(\mathbf{S}|K) = \left(\frac{1}{K}\right)^N.$$

Rather than sample the dictionaries, we will integrate them out exactly, taking advantage of properties of the Dirichlet. In particular, we have

$$\int_{\Theta} P(\mathbf{m}, \Theta|K, \mathbf{S}) = \prod_{k \in \{1:K\}} \frac{B(\alpha_k + \mathbf{n}_k)}{B(\alpha_k)}$$

where  $B(\cdot)$  is the Beta function and  $\alpha_k$  and  $\mathbf{n}_k$  are both vectors of size  $L$ , representing respectively the Dirichlet prior counts and the observed counts of each token in the dictionary.

### 3.2 Gibbs sampling for the EMM

Basic Gibbs sampling samples one variable at a time, which means, in our entity/mention model, sampling a new entity assignment  $S_n$  for some mention  $m_n$ , conditioned on all other current assignments  $\mathbf{S}_{-n}$  of mentions to entities. The initial assignment is chosen at random, then the Gibbs sampler is repeated for  $I$  iterations, each cycling through all the mentions; see Algorithm 1.

The probability  $P(S_n = k|K, \mathbf{S}_{-n}, \mathbf{m})$  is calculated by:

$$P(S_n = k|K, \mathbf{S}_{-n}, \mathbf{m}) \propto \frac{\alpha_{k,l} + n_{k,l}}{\sum_{l' \in L} (\alpha_{k,l'} + n_{k,l'})}$$

where  $\alpha_{k,l}$  and  $n_{k,l}$  are respectively the prior counts for the token  $l$  (the mention  $m_n = l$ ) and observed counts for

---

### Algorithm 1 Gibbs sampling for an entity/mention model

---

```

1: procedure GIBBS SAMPLING
2:   for n=1 to N do
3:     Sample  $S_n$  u.a.r. from  $\{1, \dots, K\}$ 
4:   end for
5:   for i=1 to I do
6:     for n=1 to N do
7:       Sample  $S_n$  from  $P(S_n|K, \mathbf{S}_{-n}, \mathbf{m})$ 
8:     end for
9:   end for
10: end procedure

```

---

token  $l$  assigned to  $E_k$ . It is divided by the sum of the prior and observed counts for all the  $L$  tokens.

Each step of the Gibbs sampler is relatively easy to compute, but of course the algorithm cannot change the number of entities; moreover, it tends to get stuck on local maxima because of the local nature of the changes [Celex *et al.*, 2000]. Despite these drawbacks, Gibbs steps are an important element used in association with split-merge steps in order to optimize the allocation of mentions to entities.

### 3.3 Split–merge MCMC for the EMM

One way to explore states with different numbers of entities is to use *birth* and *death* moves. A birth moves creates a new entity with no mentions, while a death move kills off an entity that has no mentions. While such moves, combined with Gibbs moves, do connect the entire state space, they lead to very slow mixing because death moves can only occur when Gibbs moves have removed all the mentions from an entity, which is astronomically unlikely when  $N$  is much larger than  $K$ —unless the entity being killed is one that was just born.

Split and merge moves simultaneously change the number of entities and change the assignments of multiple mentions in one go. The simplest approach is to pick an entity at random and split it into two, randomly assigning the mentions to the two new entities; the merge move operates in reverse by picking two entities and merging into one, along with their mentions. A naive implementation of this idea often fails to work, because random splits often yield a state with a very low probability, preventing acceptance. Pasula *et al.* [2003] suggested a *random mixing* procedure that chooses two entities and randomly assigns each of their mentions to one of two new entities. A split occurs when the two chosen entities happen to coincide, and a merge happens when one of the new entities receives no mentions and is discarded. The approach was effective for medium-sized data sets in their experiments (300-400 mentions, 60-80 entities) but fails when each entity has many mentions: merges become exponentially unlikely to be proposed.

Jain and Neal [2004] proposed a Restricted Gibbs Split–

Merge (RGSM) algorithm to generate splits that are consistent with data. Two random elements are chosen in the beginning. A split is proposed if these two elements belong to the same component and otherwise a merge is proposed. To split the component  $c_k$ , RGSM algorithm first assigns the elements randomly into two new components  $c_1^{launch}$  and  $c_2^{launch}$  as the launch state. Restricted Gibbs is then applied for  $t$  times inside the launch state, re-assigning elements to one of the components. The modified launch state after  $t$  Restricted Gibbs steps is used for generating the split. The resulting split reflects the data to some extent and tends to have a higher likelihood. However, these intermediate Restricted Gibbs steps are rather computationally expensive, especially for large data sets. Dahl [2003] proposed an allocation procedure, which works by assigning elements sequentially to two components. It starts with creating two new components  $c_1$  and  $c_2$  with two random elements. The remaining elements are sequentially allocated to either  $c_1$  or  $c_2$  using the Restricted Gibbs sampler conditioned on those previously assigned elements. This procedure is more efficient than preparing the launch state in RGSM.

Split–merge has been applied to different models such as the Beta Process Hidden Markov Model [Hughes *et al.*, 2012] and the Hierarchical Dirichlet Process [Wang and Blei, 2012; Rana *et al.*, 2013]; on the other hand, the split–merge method itself has not been improved since RGSM was first proposed in 2004. In the next section, we examine the interaction of the MH algorithm with split–merge moves and propose a new combination of moves that seems to work better.

## 4 SMART AND DUMB PROPOSALS

In general, smart proposals that lead to high-probability states are preferred, as they lead to faster convergence of MCMC. What Jain and Neal [2004] did for RGSM is to avoid the low-probability states generated by random splits. As we mentioned in Section 1, “smart” proposals propose states with higher probability according to the target distribution. The Gibbs sampler is smart in this sense, because its proposal distribution is proportional to the state probability and hence the MH acceptance probability is always 1 (Eq. 2). Consequently, we may instinctively conclude that the convergence efficiency might be significantly improved if we concentrate on the design of smart proposals. However, this is not true for MH in general. The MH ratio in the case of a smart merge proposal will be analyzed as an example.

Let  $q(\mathbf{x}'|\mathbf{x})$  and  $q(\mathbf{x}|\mathbf{x}')$  be respectively a smart merge proposal and a smart split proposal. Each first picks a subset of the variables to merge (or split) with probability  $P_m$  (or  $P_s$ ). It then proposes a particular merge (or split) according to the target distribution  $f_m$  (or  $f_s$ ), where  $f_m$  and  $f_s$  are

proportional to state probabilities:

$$f_m = \frac{\pi(\mathbf{x}')}{\sum_{\omega \in W(\mathbf{x})} \pi(\omega)}, f_s = \frac{\pi(\mathbf{x})}{\sum_{\omega \in W(\mathbf{x}')} \pi(\omega)}, \quad (3)$$

where  $W(\mathbf{x})$  and  $W(\mathbf{x}')$  are respectively the set of states for all possible merges and the set of states for all possible splits given  $P_s$  and  $P_m$ .

The MH ratio is then given by

$$\begin{aligned} \frac{q(\mathbf{x}|\mathbf{x}') \pi(\mathbf{x}')}{q(\mathbf{x}'|\mathbf{x}) \pi(\mathbf{x})} &= \frac{P_s \frac{\pi(\mathbf{x})}{\sum_{\omega \in W(\mathbf{x}')} \pi(\omega)} \pi(\mathbf{x}')}{P_m \frac{\pi(\mathbf{x}')}{\sum_{\omega \in W(\mathbf{x})} \pi(\omega)} \pi(\mathbf{x})} \\ &= \frac{P_s \sum_{\omega \in W(\mathbf{x})} \pi(\omega)}{P_m \sum_{\omega \in W(\mathbf{x}')} \pi(\omega)} \end{aligned} \quad (4)$$

When both split and merge are smart, the ratio  $\frac{P_s}{P_m}$  will be very low due to a quite small  $P_s$ . It is because that the smart split would not give a big probability mass to the part a smart merge prefers to merge (detailed examples given in the next subsection). The second part in Formula 4, namely space ratio, is important in split–merge case because of the space asymmetry.  $\sum_{\omega \in W(\mathbf{x})}$  on the top is just  $\pi(\mathbf{x})$  since when we have chosen two entities to merge ( $P_m$ ), there is only one merge possibility. However,  $\sum_{\omega \in W(\mathbf{x}')}$  contain  $2^n$  possible splits ( $n$  is the number of mentions assigned to the picked entity).

The  $\frac{P_s}{P_m}$  ratio gives the first idea about why smart proposals have conflicts with inverse smart proposals. In case of space asymmetry, a smart–smart proposal suffers also from the space ratio in addition to the  $\frac{P_s}{P_m}$  ratio.

### 4.1 Why smart proposals do not work by themselves: A simple example

Let’s take a concrete example of split and merge to illustrate the problem stated above for smart proposals. Assuming that there are three entities in state  $\mathbf{x}$  as below:

$$\mathbf{x} : \{\mathbf{E}_1 : \{A A B B\}; \mathbf{E}_2 : \{C C\}; \mathbf{E}_3 : \{C C\}\},$$

a desired split would be splitting  $\mathbf{E}_1$  into two entities as follows:

$$\mathbf{x}'_1 : \{\mathbf{E}_1 : \{A A\}; \mathbf{E}_4 : \{B B\}; \mathbf{E}_2 : \{C C\}; \mathbf{E}_3 : \{C C\}\}.$$

A smart split proposal distribution should propose the state  $\mathbf{x}'_1$  with a quite high probability as illustrated in the left part of Figure 1, where the width of the arrow line indicates the probability value. However, a smart merge proposal, starting from state  $\mathbf{x}'_1$ , would rather have a high probability  $q(\mathbf{x}'_1|\mathbf{x}'_1)$  for proposing the state  $\mathbf{x}'_1$ :

$$\mathbf{x}''_1 : \{\mathbf{E}_1 : \{A A\}; \mathbf{E}_4 : \{B B\}; \mathbf{E}_2 : \{C C C C\}\}.$$

From the point of view of a smart merge, inverting the smart split’s preferred move  $q(\mathbf{x}|\mathbf{x}'_1)$  is highly unlikely, as

represented by the dashed arrow in the figure. Therefore, considering the high value of  $q(\mathbf{x}'_1|\mathbf{x})$ , the proposal ratio becomes extremely low, which leads to a very low acceptance rate for smart split proposals.

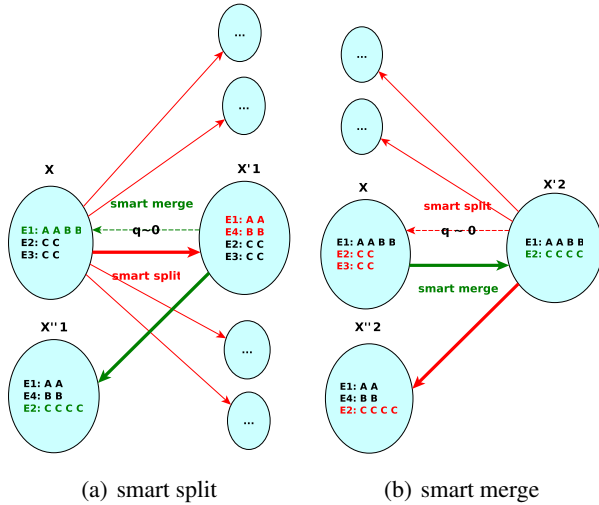


Figure 1: Conflicts between smart split and smart merge (red lines for splits and green lines for merges; thick/dashed lines for moves preferred/dispreferred by smart proposals.)

It is a similar situation for smart merge proposals, as illustrated in the right part of Figure 1. A smart merge proposal will give a high probability to generate the state  $\mathbf{x}'_2$ :

$$\mathbf{x}'_2 : \{\mathbf{E}_1 : \{A A B B\}; \mathbf{E}_2 : \{C C C C\}\}.$$

On the other hand, the smart split from the state  $\mathbf{x}'_2$  will be more likely to generate a state such as:

$$\mathbf{x}''_2 : \{\mathbf{E}_1 : \{A A\}; \mathbf{E}_3 : \{B, B\}; \mathbf{E}_2 : \{C C C C\}\},$$

leaving only a tiny probability to propose the state  $\mathbf{x}'_2$  to go back to  $\mathbf{x}$ . The same phenomenon of a low acceptance rate will be engendered.

For the particular case of split and merge, a new entity created by the split proposal changes the parameters of the space, which means, for one split and its inverse merge, there are much more possibilities for splits than those for merges. This neighborhood issue makes the smart merge proposal even harder to be accepted as shown in Formula 4.

## 4.2 Coupling with smart and dumb proposals

Smart proposals are not effective in this case because the entity which we choose to split does not correspond to the entities that we prefer to merge in the reverse direction. However, if we want to distribute a higher probability to the reverse move,  $q(\mathbf{x}|\mathbf{x}'_1)$  for instance, it can be treated as a dumb proposal rather than a smart one. “Dumb” proposals can be considered as distributions that give uniform probability mass over all possible moves.

An important property of MCMC methods is that *detailed balance* can be guaranteed when several different proposals are adopted on condition that each proposal satisfies the Metropolis-Hastings algorithm [Tierney, 1994]. Therefore, there is a solution to combine smart and dumb proposals, namely the *Smart-Dumb Dumb-Smart* proposals (SDDS), as illustrated in Figure 2.

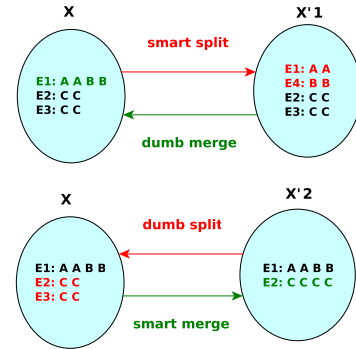


Figure 2: Smart-Dumb Dumb-Smart design for proposals

As a consequence, there are two separate pairs of proposal distributions. For either of them, the dumb proposal gives a uniform distribution over all possible moves. The existence of dumb proposals helps the acceptance of smart proposals. In the context of these two pairs, both smart split and smart merge can produce higher acceptance rates and faster mixing.

## 4.3 An SDDS split–merge algorithm

Inside the SDDS algorithm, we want to propose high-probability states with smart splits and smart merges and to do so efficiently. The algorithm for the smart split proposal with dumb merge and the one for smart merge proposal with dumb split are respectively described in Algorithm 2 and Algorithm 3.

**Smart split/dumb merge proposal** Algorithm 2 begins by choosing randomly between a smart split proposal and a dumb merge proposal. If a smart split is picked, it will first choose one entity  $E_k$  based on a function  $f_{split}(E_k)$ . The function is inversely proportional to the likelihood of the mentions  $\mathbf{m}_{E_k}$  associated with this entity  $E_k$ , which implies that the proposal tends to choose large and mixed entities. The likelihood is given by  $B(\alpha_k + \mathbf{n}_k)/B(\alpha_k)$  where  $\alpha_k$  and  $\mathbf{n}_k$  are respectively the vectors for the Dirichlet prior counts and the observed counts of each token in  $\mathbf{m}_{E_k}$ .

Once the entity  $E_k$  chosen, the smart split procedure will allocate sequentially each mention to one of two newly created entities  $E'_1$  and  $E'_2$ , according to the likelihood of the previously assigned mentions. Given the entity  $E_k : \{AABBCC\}$  for example, the procedure is illustrated in

---

**Algorithm 2** Smart Split and Dumb Merge proposal

---

- 1: **procedure** SMART SPLIT DUMB MERGE
- 2:   choose a move type:  $type \sim (split, merge)$
- 3:   **if** type==split **then** ▷ smart split
- 4:     choose one entity to split
- 5:

$$E_k \sim f_{split}(E_k)$$
$$f_{split}(E_k) \propto \frac{1}{P(\mathbf{m}_{E_k}|E_k)}$$

- 6:     create two new empty entities  $E'_1$  and  $E'_2$
  - 7:     assign each mention in  $E_k$  sequentially to  $E'_1$  or  $E'_2$  according to the likelihood of previously assigned mentions
  - 8:   **else** ▷ dumb merge
  - 9:     choose one entity uniformly from K entities
  - 10:    choose another entity uniformly from the rest K-1 entities
  - 11: **end if**
  - 12:   calculate the acceptance ratio  $\alpha$
  - 13:   apply the proposal with probability  $\alpha$
  - 14: **end procedure**
- 

Table 1.

Table 1: Splitting one entity into two entities sequentially

| steps  | 0 | 1   | 2   | 3     | 4     | 5   | 6     |
|--------|---|-----|-----|-------|-------|-----|-------|
| $E'_1$ |   | A   | AA  | AA    | AA    | AA  | AA    |
| $E'_2$ |   |     |     | B     | BB    | BBC | BCCC  |
| P      |   | 0.5 | 0.6 | 0.625 | 0.714 | 0.5 | 0.714 |

Two new created entities are empty in the beginning. During each step, the allocation probability for each mention  $m_i$  is calculated by:

$$P(E'_1|m_i) \propto \frac{\alpha_{1,l} + n_{1,l}}{\sum_{l' \in L} (\alpha_{1,l'} + n_{1,l'})}$$
$$P(E'_2|m_i) \propto \frac{\alpha_{2,l} + n_{2,l}}{\sum_{l' \in L} (\alpha_{2,l'} + n_{2,l'})} \quad (5)$$

where  $\alpha_{1,l}$  and  $\alpha_{2,l}$  are the Dirichlet priors for the token  $l$  (the mention  $m_i = l$ , which is A, B or C in this case) being assigned to entity  $E'_1$  and  $E'_2$ ,  $n_{1,l}$  and  $n_{2,l}$  are current observed counts of token  $l$  assigned to  $E'_1$  and  $E'_2$  (counts are updated during each step). The denominator sums up the prior and observed counts for all possible tokens (A, B and C in this case). The probability in each step is given in the table taking all  $\alpha_{1,l}=\alpha_{2,l}=1$  as example (smaller alpha makes this procedure more discriminating). The probability of this allocation procedure is then a product of the probability in each step. This procedure avoids the time-consuming Restricted Gibbs sampling adopted by Jain and Neal [2004]. [Dahl, 2003] proposed a similar sequential procedure but started by creating two new entities with two

---

**Algorithm 3** Smart Merge and Dumb Split proposal

---

- 1: **procedure** SMART MERGE DUMB SPLIT
- 2:   choose a move type:  $type \sim (split, merge)$
- 3:   **if** type==merge **then** ▷ smart merge
- 4:     choose one entity  $E_i$  uniformly from K entities
- 5:     choose another entity

$$E_j \sim f_{merge}(E_j|E_i)$$

$$f_{merge}(E_j|E_i) \propto P(\mathbf{m}_{E_i, E_j}, E_j|E_i)$$

- 6: **else** ▷ dumb split
  - 7:   choose one entity  $E_k$  uniformly from K entities
  - 8:   create two new empty entities  $E'_1$  and  $E'_2$
  - 9:   assign each mention in  $E_k$  sequentially to  $E'_1$  or  $E'_2$  with equal probability
  - 10: **end if**
  - 11:   calculate the acceptance ratio  $\alpha$
  - 12:   apply the proposal with probability  $\alpha$
  - 13: **end procedure**
- 

random mentions, which causes an initial bias when these two random mentions are supposed to be associated with the same entity.

The reverse dumb merge proposal would rather generate random merges. It picks one entity uniformly from K entities and then picks another from the remaining K-1 entities. The probability of this reverse proposal is then  $2/K(K-1)$  (two orders of choosing these two entities).

If a dumb merge proposal is chosen in the beginning, the choice of two entities will have the probability  $1/K(K-1)$ . Then we need to know the probability of the split proposal that reverses this move, i.e., allocates the mentions exactly into the two given sets. The order of mentions during allocation influences the final probability and the different probabilities from all possible orders are supposed to be summed up, which is not really feasible in practice. Dahl [2003] applied a random permutation on the order of mentions, which may be critical to correctness of MCMC methods since in this way we are obtaining a random probability for the reverse split when merge is proposed and it may not correspond to the exact probability of the inverse move. In our case, we fix a unique order for all mentions so that there is only one way of applying the procedure thus only one possible probability value for the same split results, either the real split procedure or the imaged reverse one. This unique order is chosen in an arbitrary way. The choice of any particular order has no influence on the inference but the same order should be kept all along the experiments.

**Smart merge/dumb split proposal** Algorithm 3 begins from making the random choice between merge and split as well. If smart merge proposal is picked, it will first choose

one entity  $E_i$  randomly from  $K$  entities. The choice of the second entity  $E_j$  is based on how likely it is when merged with  $E_i$ . The entity  $E_j$  is drawn from a distribution given by function  $f_{merge}(E_j|E_i)$ , which is proportional to the likelihood of this resulting merge of  $E_j$  to  $E_i$ . The likelihood of the merge is calculated by:

$$P(\mathbf{m}_{E_i, E_j}, E_j|E_i) = \frac{B(\boldsymbol{\alpha}_i + \mathbf{n}_i + \mathbf{n}_j)}{B(\boldsymbol{\alpha}_i)} \quad (6)$$

where  $\mathbf{m}_{E_i, E_j}$  refers to all mentions assigned to  $E_i$  and  $E_j$ ,  $\boldsymbol{\alpha}_i$  is vector for Dirichlet prior, and  $\mathbf{n}_i + \mathbf{n}_j$  refers to the vector for the observed counts of each term in  $\mathbf{m}_{E_i, E_j}$ .

The reverse dumb split proposal chooses one entity  $E_k$  uniformly from  $K$  entities and allocates each mention randomly into two new created entities, the probability of which is  $1/(K \cdot 2^n)$ , where  $n$  is the number of mention assigned to entity  $E_k$ .

If the dumb split proposal is chosen, it generates a random split with the same probability  $1/(K \cdot 2^n)$ . As for the reverse smart merge proposal, we need to consider the merge in two different orders, namely  $f_{merge}(E_j|E_i)$  and  $f_{merge}(E_i|E_j)$ , where  $f_{merge}(E_i|E_j)$  is proportional to  $P(\mathbf{m}_{E_j, E_i}, E_i|E_j)$  which can be calculated similarly to Formula 6.

## 5 EXPERIMENTS WITH SDDS

### 5.1 Applying SDDS to EMM

We applied the SDDS algorithm to the entity/mention model. During each inference step, the SDDS sampler chooses uniformly from either smart-split/dumb-merge proposal or dumb-split/smart-merge proposal; this is then complemented by one single Gibbs sampling step.<sup>2</sup> It is worth emphasizing again that detailed balance is satisfied when each sub-kernel fulfills the detailed balance condition individually. Three other algorithms are tested for comparison. The first is a random mixing (RM) sampler inspired from [Pasula *et al.*, 2003]. It works by choosing two random entities (which could be the same one) and then distributing corresponding mentions into two new created entities by a randomly chosen split point. The second is the RGSM sampler [Jain and Neal, 2004].<sup>3</sup> The third is a smart-smart (SS) sampler which has the same smart split and merge moves as SDDS but lacks the paired dumb moves.

The simulated data sets use 10 different tokens (A, B, C, etc.). Different configurations were tested, including dif-

<sup>2</sup>The different ways of combining a designed sampler with Gibbs sampler is an issue to be investigated; we adopted this configuration for all tested algorithms for comparison.

<sup>3</sup>The stable version (5, 1, 1) is adopted for comparison, which contains 5 intermediate Restricted Gibbs steps inside one MH step and then one complete Gibbs sampling.

ferent data set sizes ( $N=100, 200, 500$ ) and different initial entity numbers ( $K_0=1, 5, 10, 20$ ). All Dirichlet priors  $\alpha$  are set to 0.001. For the Log-normal prior on the number of entities  $K$ , the location parameter is set to 0.5 (10 entities) and scale parameter is set to 1. Experiments were run for 10K/50K/200K iterations, with a time-out at 10 hours.

We are interested in the posterior distribution of  $K$ , the number of entities, given the available evidence. We illustrate the evolution of the expected value of  $K$  for the various algorithms as a function of the number of iterations, for different values of  $N$  and  $K_0$ . We also give the acceptance rates and time needed for each iteration for comparison.

**Posterior distribution of entity numbers** The posterior distributions of entity numbers are analyzed by the mean of the value  $K$  during iterations. First of all, we fix the initial value  $K_0=5$  to compare results of these four different algorithms for different data sizes, as shown in Figure 3. We can see that for  $N = 100$  and  $N=200$ , all samplers except random mixing sampler can converge to the correct posterior  $K=10$ . However, when a larger data set  $N=500$  is used, the RGSM sampler fails to have successful split or merge therefore the value  $K$  is trapped at initial value. The SS sampler is capable of applying several effective splits whereas the merge proposal does not work well because of the reason we discussed in Section 4.1.

For the data set  $N=500$ , we have also analyzed the sensitivity of the sampler to initial value  $K_0$ , as illustrated in Figure 4. It is easy to observe that RGSM sampler is always trapped in initial  $K_0$  in this case. The SS sampler can generate well-assigned mentions for entities during smart split procedure, whereas it can not converge to the true posterior. It is interesting to see from the results of Random Mixing sampler that it works well when the initial value is twice as large as the true value. In fact, in the RM sampler, the random split procedure encourages the acceptance of merge proposals. However, the random split proposals themselves are not likely to be accepted because they generate very low-probability states. The SDDS sampler outperforms all the others, converging to the true posterior regardless of the initial value.

It is worth mentioning that the inferences start from random allocations, which would generate an ensemble of messed up entities. It is not really useful to merge messed up entities. The acceptance rates for merging messed up entities should be very low as well. It is therefore logical to observe many accepted splits in the beginning of the inference of SDDS algorithm. SDDS algorithm generates more well-formed entities and then yields higher probabilities for merging them.

**Acceptance rates** The relation between the size of data set and the acceptance rates for both split and merge are shown in Figure 5. We can observe that, for the data set

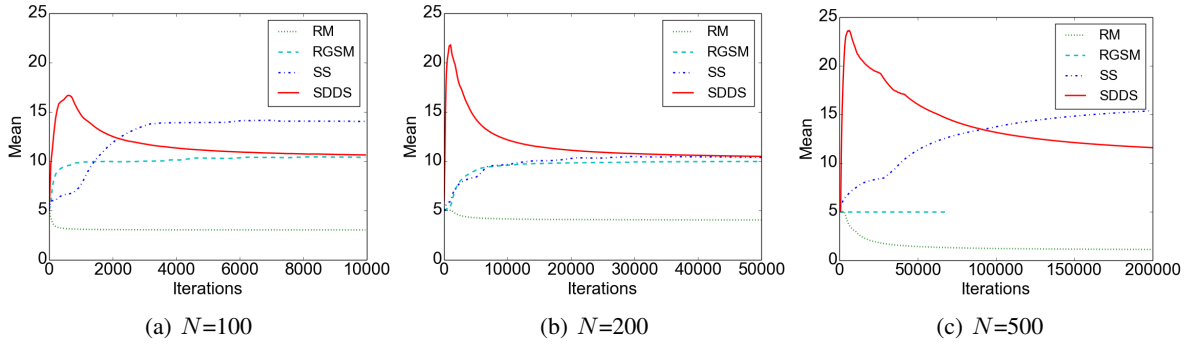


Figure 3: The mean of  $K$  during iteration for different data size, with the initial  $K_0=5$

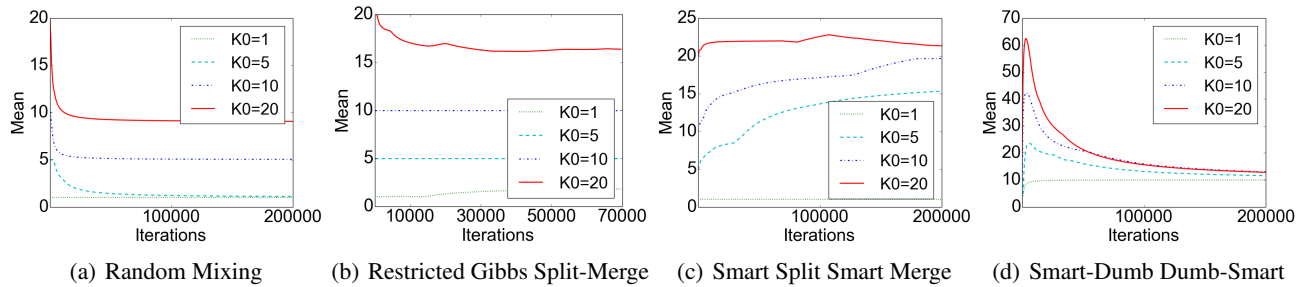


Figure 4: The mean of  $K$  during iteration for different initial  $K_0=1,5,10,20$ , with the data size  $N=500$

$N=100$ , RGSM sampler has high acceptance rates for both split (7.8%) and merge (2.1%). However, when the data size is  $N=200$ , the acceptance rates are decreased largely, only 1.1% for split and 0.08% for merge. When the data size grows to  $N=500$ , almost no effective split or merge is happening. On the contrary, the smart proposals in SDDS algorithm, either smart split or smart merge, maintain satisfying acceptance rates even for the data set  $N=500$ , 1.9% for smart split and 4.8% for smart merge. For the SS sampler, the drop of acceptance rates is similar to RGSM sampler since there is no dumb proposals to support their corresponding smart proposals.

**Time per iteration** As we stated previously, the allocation procedure in our smart split proposal is less time-consuming than the Restricted Gibbs sampling based split proposal. The performance of the running time per iteration is shown for each algorithm in Figure 5. For the RGSM sampler, the time spent per iteration grows quickly with the data size, whereas the time for SDDS algorithm remains stable. In the case of  $N=500$ , RGSM sampler takes 488.63 milliseconds per step while SDDS algorithm takes only 10.02 milliseconds per step. The time per iteration for Random Mixture sampler and that for SS sampler (which are not show in the figure) is in the same scale of SDDS sampler and stays stable for different data sizes.

## 5.2 Applying SDDS to conjugate Dirichlet Process Mixture Model

RGSM algorithm is originally proposed for Dirichlet Process Mixture Model (DPMM). When conjugate priors are used, the Gibbs sampling procedure can be easily constructed for DPMM. A particular Gibbs sampling method is adapted in this context of DPMM model so that the Gibbs sampler can create new components [Neal, 1992]. The proposed SDDS algorithm is also applied to DPMM and is then compared to the Gibbs sampler and RGSM sampler.

The experiments have been done with high dimensional Bernoulli data. Given the independently and identically distributed data set  $\mathbf{y} = (y_1, y_2, \dots, y_N)$ , each observation  $y_i$  has  $m$  Bernoulli attributes,  $(y_{i1}, y_{i2}, \dots, y_{im})$ . Given the component  $c_i$  each item  $y_i$  belongs to, its attributes are independent of each other. The mixture components are considered as the latent class that produces the observed data.

The simulated data for our experiments are generated in the same way as Jain and Neal [2004] did for one high-dimensional data set: 5 components with attribute dimension 15. Experiments are run for different sizes,  $N=100, 1K, \text{ and } 10K$ . The Dirichlet process prior and the Beta prior for attributes are respectively set to 1 and 0.1. (See Jain and Neal [2004] for further details on this model).

Jain and Neal [2004] have demonstrated their results by plotting the traces of the five highest-weight components

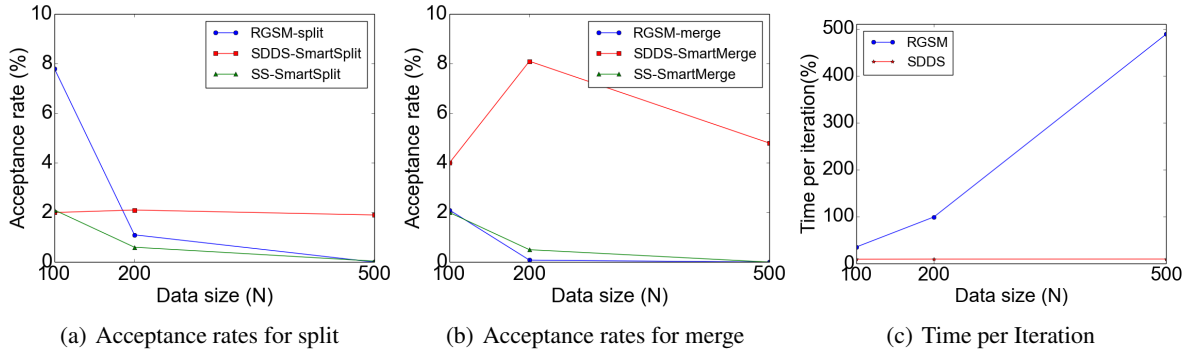


Figure 5: The differences of acceptance rates and time per iteration for different data sizes

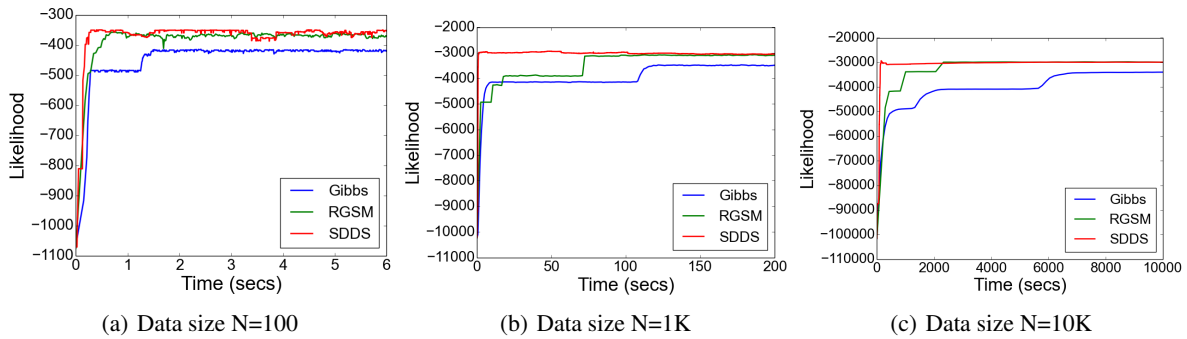


Figure 6: Comparison of the evolution of likelihood during the time for different data sizes

to verify if their algorithms can cover most data and detect five components. We have provided the same plots for the SDDS sampler. We observed the same relative performance for SDDS and RGSM on the DPMM as for the EMM, in terms of time per iteration and acceptance rates (results not shown here). We also compared the evolution of the likelihoods for SDDS, RGSM, and Gibbs as a function of running time (Figure 6). We see that, for  $N=100$ , RGSM arrives at high-probability states about 0.5 second after the SDDS algorithm does. When  $N=1K$ , SDDS gains 70 seconds over RGSM. When  $N=10K$ , SDDS outperforms RGSM by more than 2000 seconds. We conclude from this limited sample of runs that the advantage of SDDS over RGSM increases with data set size.

## 6 CONCLUSION

We have described the SDDS algorithm, which achieves efficient split-merge inference by combining smart and dumb proposals. The idea is illustrated for the entity/mention model. For the smart split proposal, we proposed a fast and exact way of generating splits that are consistent with data. Experiments on the entity/mention model and Dirichlet process mixture models suggest that SDDS algorithm mixes faster than previously known algorithms, and that the advantage increases with data set size.

## References

- Gilles Celeux, Merrilee Hurn, and Christian P. Robert. Computational and inferential difficulties with mixture posterior distributions. *Journal of the American Statistical Association*, 2000.
- J Chang and J. W. Fisher. Parallel sampling of dp mixture models using sub-clusters splits. In *NIPS*, 2013.
- David B Dahl. An improved merge-split sampler for conjugate dirichlet process mixture models. Technical report, University of Wisconsin - Madison, 2003.
- Andrew Gelman. Iterative and non-iterative simulation algorithms. In *Computing Science and Statistics: Proceedings of the 13th Symposium on the Interface*, 1992.
- Stuart Geman and Donald Geman. Stochastic relaxation, gibbs distributions, and the Bayesian restoration of images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 1984.
- W K Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 1970.
- Michael Hughes, Emily Fox, and Erik Sudderth. Effective split-merge Monte Carlo methods for nonparametric models of sequential data. In *NIPS*, 2012.
- Sonia Jain and Radford Neal. A split-merge Markov chain Monte Carlo procedure for the dirichlet process mixture



- model. *Journal of Computational and Graphical Statistics*, 2004.
- Zia Khan, T. Balch, and F. Dellaert. Multitarget tracking with split and merged measurements. In *CVPR*, 2005.
- N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 1953.
- Radford M. Neal. Bayesian mixture modeling. In *Proceedings of the 11th International Workshop on Maximum Entropy and Bayesian Methods of Statistical Analysis*, 1992.
- Mathias Niepert and Pedro Domingos. Exchangeable variable models. In *ICML*, 2014.
- Hanna Pasula, Stuart Russell, Michael Ostland, and Ya'acov Ritov. Tracking many objects with many sensors. In *IJCAI*, 1999.
- Hanna Pasula, Bhaskara Marthi, Brian Milch, Stuart Russell, and Ilya Shpitser. Identity uncertainty and citation matching. In *NIPS*. MIT Press, 2003.
- Santu Rana, Dinh Phung, and Svetha Venkatesh. Split-merge augmented gibbs sampling for hierarchical dirichlet processes. In *Advances in Knowledge Discovery and Data Mining*. Springer, 2013.
- Luke Tierney. Markov chains for exploring posterior distributions. *Annals of Statistics*, 1994.
- Chong Wang and David M. Blei. A split-merge MCMC algorithm for the hierarchical dirichlet process. *CoRR*, 2012.
- Sinead Williamson, Avinava Dubey, and Eric P. Xing. Parallel Markov chain Monte Carlo for nonparametric mixture models. In *ICML*, 2013.

---

# Planning under Uncertainty with Weighted State Scenarios

---

**Erwin Walraven**  
Delft University of Technology  
Mekelweg 4, 2628 CD  
Delft, The Netherlands

**Matthijs T. J. Spaan**  
Delft University of Technology  
Mekelweg 4, 2628 CD  
Delft, The Netherlands

## Abstract

In many planning domains external factors are hard to model using a compact Markovian state. However, long-term dependencies between consecutive states of an environment might exist, which can be exploited during planning. In this paper we propose a *scenario* representation which enables agents to reason about sequences of future states. We show how weights can be assigned to scenarios, representing the likelihood that scenarios predict future states. Furthermore, we present a model based on a Partially Observable Markov Decision Process (POMDP) to reason about state scenarios during planning. In experiments we show how scenarios and our POMDP model can be used in the context of smart grids and stock markets, and we show that our approach outperforms other methods for decision making in these domains.

## 1 INTRODUCTION

The Markov Decision Process (MDP) formalism is a mathematical framework for modeling agents interacting with their environment (Puterman, 1994). In many real-world planning domains, however, external factors can be difficult to predict, which makes it hard to obtain a Markovian model with the right state features and an appropriate level of detail (Witwicki et al., 2013). In such domains, it is hard to estimate probabilities for the occurrence of uncertain events, and therefore decision making can be a challenging task.

An example of a hard-to-model external factor is renewable energy supply such as generation of wind power. Research has shown that the most severe problems in electricity grids occur during peak-load hours when energy demand is high and wind power generation is interrupted (Moura and De Almeida, 2010), because then the supply of renewable elec-

tricity may not be sufficient to satisfy the demand of consumers. A potential solution is exploiting the flexibility of the loads of consumers, such that they can be supplied during off-peak hours. This solution requires reasoning about future wind speed, but many external factors influencing wind make it hard to define a compact Markovian state for wind. Additionally, methods to predict short-term wind power are affected by errors and may be inaccurate (Giebel et al., 2011).

In order to accommodate planning in domains with events that are difficult to predict and hard to model, we propose a framework that enables agents to reason about future states. This approach is based on the observation that there can be long-term dependencies between states, which can be exploited during planning, rather than explicitly defining a state transition model with appropriate features. In our framework, such long-term dependencies are modeled by scenarios, which are sequences of states. An advantage of using scenarios is illustrated in Figure 1, in which we compare wind predictions generated by a second-order Markov chain and actual wind scenarios that have been observed in practice. The lines in the figure visualize the 5th percentile, mean and 95th percentile of these predictions, and show us that scenarios provide information that is not sufficiently modeled by a second-order Markov chain.

In our work we assign weights to scenarios, corresponding to the likelihood that a scenario predicts future states accurately, and we use the Partially Observable Markov Decision Process framework (Kaelbling et al., 1998) to reason about scenarios during planning. We demonstrate the proposed Scenario-POMDP model in two domains. Besides wind scenarios in smart grids, we show how a Scenario-POMDP can be applied to financial stock markets. This domain has also been subject of study in the artificial intelligence community, since stock price is hard to model and depends on many external factors (Hassan and Nath, 2005). An experimental evaluation shows that our method outperforms other methods for decision making in both domains, indicating that scenarios are a valuable representation to model uncertainty regarding the future.

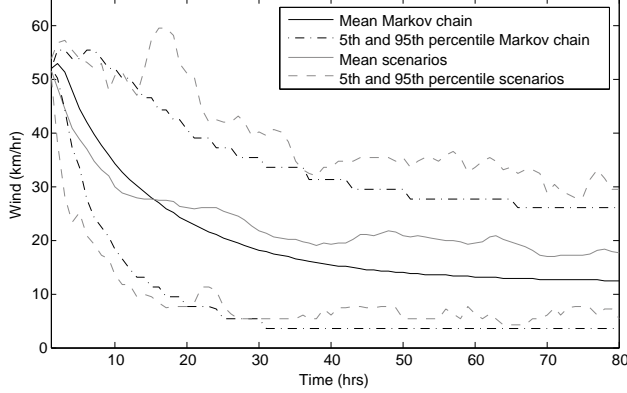


Figure 1: Comparison between Markov chain wind predictions and realistic wind scenarios starting from 51 km/hr.

The structure of this paper is as follows. Section 2 introduces planning under uncertainty. In Section 3 we introduce scenarios and related concepts. In Sections 4 and 5 we show how planning with scenarios can be applied to smart grids and stock markets. In the remaining sections we discuss related work, conclusions and future work. In the supplementary material we provide problem formulations and additional information regarding the problem domains.

## 2 PLANNING UNDER UNCERTAINTY

Planning under uncertainty involves agents that interact with their environment by executing actions, and observing effects caused by these actions. This is a challenging problem if agents are uncertain about the outcome of their action execution, and if they cannot fully observe the environment they are acting in. The Partially Observable Markov Decision Process (POMDP) formalism provides a framework to plan in such uncertain environments (Kaelbling et al., 1998).

In a POMDP, it is assumed that the environment is in a state  $s \in S$ . After executing an action  $a \in A$  in state  $s$ , the state of the environment transitions to another state  $s' \in S$  according to probability distribution  $P(s'|s, a)$  and a reward  $R(s, a)$  is received from the environment. A state transition from  $s$  to  $s'$  is only conditionally dependent on state  $s$  and action  $a$ , which is called the Markov property. In contrast to MDPs with full observability (Puterman, 1994), the agent does not directly perceive the state of the environment in a POMDP. It receives an observation  $o \in O$  that can be used to reason about the underlying MDP state of the environment, using a probability distribution  $P(o|a, s')$ . Since states are not directly observable in a POMDP, agents maintain a belief state, denoted  $b$ , which represents a probability distribution over states. The resulting belief state  $b_a^o$  after executing action  $a$  and observing  $o$

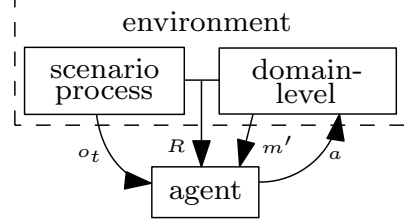


Figure 2: An agent executing action  $a$ , which causes a state transition to  $m'$ , and the agent receives state observation  $o_t$  from the scenario process at time  $t$  and receives reward  $R$ .

in belief state  $b$  can be determined using Bayes' rule:

$$b_a^o(s') = \frac{P(o|a, s')}{P(o|a, b)} \sum_{s \in S} P(s'|s, a)b(s)$$

where  $P(o|a, b) = \sum_{s' \in S} P(o|a, s') \sum_{s \in S} P(s'|s, a)b(s)$ . The state space  $S$ , action space  $A$  and observation space  $O$  are assumed to be finite in this paper.

To act in a partially observable environment, agents use a policy  $\pi(b)$ , which maps belief states to actions. A policy  $\pi(b)$  is characterized by a value function  $V^\pi(b)$  defining the expected discounted reward collected by the agent when executing policy  $\pi$  from belief state  $b$ . The optimal value function  $V^*(b)$  is defined as:

$$V^*(b) = \max_{a \in A} \left[ \sum_{s \in S} R(s, a)b(s) + \gamma \sum_{o \in O} P(o|a, b)V^*(b_a^o) \right]$$

where  $b_a^o$  is defined by Bayes' rule, and  $\gamma$  is a discount factor satisfying  $0 \leq \gamma < 1$ . Computing exact solutions to POMDPs is known to be intractable (Papadimitriou and Tsitsiklis, 1987), but many approximate methods exist based on point-based value iteration (Pineau et al., 2003; Spaan and Vlassis, 2005). In this paper we use POMCP (Silver and Veness, 2010), an online Monte-Carlo planning algorithm that is capable of dealing with a large number of states. We assume the planning horizon to be finite.

## 3 PLANNING WITH SCENARIOS

In this section we propose a scenario representation for planning under uncertainty. First we introduce the notion of scenarios and we explain how scenarios can be weighted based on previous observations. We also present a general POMDP model to reason about scenarios and future states during planning.

### 3.1 SCENARIOS AND WEIGHTS

We assume that an agent interacts with an environment as shown in Figure 2. The environment consists of a process

**input** : observation sequence  $o_{1,t}$ , scenario set  $X$ ,  
threshold  $\rho$

**output**: weights  $w$

$X' \leftarrow \emptyset$

$d \leftarrow 0$

**while**  $|X'| < \rho$  **do**

$X' \leftarrow \{x \in X : W(x, o_{1,t}) \leq d\}$

$d \leftarrow d + 1$

**end**

**foreach**  $x \in X$  **do**

**if**  $x \in X'$  **then**

$w_x \leftarrow 1 / (\varepsilon + W(x, o_{1,t}))$

**else**

$w_x \leftarrow 0$

**end**

**end**

$w^* \leftarrow \sum_{x \in X} w_x$

**foreach**  $x \in X$  **do**

$w_x \leftarrow w_x / w^*$

**end**

**Algorithm 1: WEIGHTS.**

for which the domain-level state changes to  $m'$  after executing an action. For simplicity, in this paper we assume that the state of this process is observable, but our approach is not limited to this assumption. Additionally, there is another process for which a compact Markovian model does not exist, called the scenario process. We assume that an agent observes a numerical-valued state  $o_t$  of this process at time  $t$ , which we call a state observation, but there is no model available defining the state transitions. The actions executed by the agent do not influence the state transitions of the scenario process. We assume that the rewards received by the agent depend on the state  $m$ , as well as on the state of the scenario process, which means that the agent has to account for future states to optimize the long-term reward.

In order to be able to reason about future states, we propose a scenario representation below. A scenario is a sequence of states of the scenario process, and implicitly models the dependencies between multiple consecutive states.

**Definition 1.** (*Scenario*). A scenario  $x = (x_1, \dots, x_T)$  is a sequence of states of the scenario process for  $T$  consecutive timesteps, where  $x_t$  is the state at time  $t$ . The sequence containing the first  $t$  states of scenario  $x$  is denoted by  $x_{1,t}$ .

States of the scenario process are assumed to be directly observable, represented by a sequence  $o_{1,t} = (o_1, o_2, \dots, o_t)$ , containing state observations from the first  $t$  timesteps. The sequence of state observations can be compared to a scenario, by comparing the individual state observations with states in the scenario. We illustrate this with a small example for the scenario  $x = (x_1, x_2, x_3, x_4) = (8, 5, 3, 2)$ . Suppose that the state observations are defined by the se-

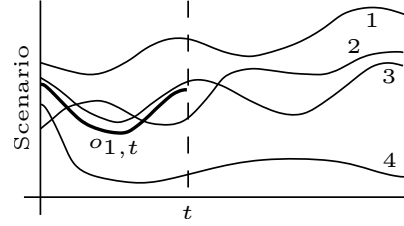


Figure 3: Scenarios and state observations until time  $t$ .

quence  $(o_1, o_2, o_3) = (8, 5, 4)$ , then the first and second state observation are identical to the first two states defined by scenario  $x$ , and the third state observation is different.

Weights can be assigned to scenarios, representing the likelihood that a scenario perfectly predicts the states that will be observed in the future. To reason about future states, we assume that a large scenario set  $X$  is given, containing sequences of states that can be observed in  $T$  consecutive timesteps. If a sequence of states  $o_{1,t}$  is observed until time  $t$ , then the sequence can be used to assign weights to the scenarios in  $X$ . In Sections 4 and 5 we discuss how such a scenario set can be obtained in realistic domains.

An informal visual representation of scenarios is shown in Figure 3. It shows four scenarios, labeled 1 to 4, and the state observation sequence  $o_{1,t}$ . As can be seen in the figure, the state observation sequence does not correspond to any scenario until time  $t$ , but it is very similar to scenario 3. If  $X$  is an accurate set of scenarios, then it is probable that scenario 3 predicts future states. Therefore, the weight assigned to this scenario should be high in comparison to the weights assigned to other scenarios.

The weights assigned to scenarios are inversely proportional to the distance between scenarios and the state observation sequence. The distance between state observation sequence  $o_{1,t} = (o_1, o_2, \dots, o_t)$  and the first  $t$  states of a scenario  $x \in X$  can be measured by computing the sum of squared errors. The function  $W$  below computes this distance for a given scenario  $x \in X$  and state observation sequence  $o_{1,t}$ :

$$W(x, o_{1,t}) = \sum_{i=1}^t (o_i - x_i)^2.$$

We select scenarios up to a certain distance from the state sequence until time  $t$ , and we assign weights to the scenarios such that they sum to 1. The algorithm that we use to assign weights is shown in Algorithm 1. It selects a subset of scenarios  $X'$  containing at least  $\rho$  scenarios similar to  $o_{1,t}$ , based on the sum of squared errors. This step ensures that there is a sufficient number of scenarios with non-zero weight, to prevent that the future is predicted by only one or very few scenarios. A probability distribution is defined over the scenarios in  $X'$ , in which the probabilities are inversely proportional to the computed distance. A nor-

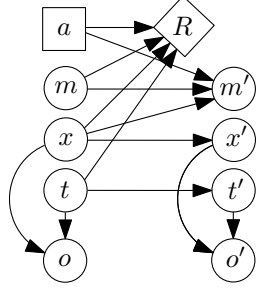


Figure 4: General dynamic Bayesian network of the Scenario-POMDP model.

malization step is performed to ensure that the sum of the probabilities equals 1. The parameter  $\rho$  can be adjusted to lower-bound the number of scenarios with non-zero probability.

### 3.2 SCENARIO-POMDP MODEL

In this section we present a POMDP model, which we can use to model any planning problem with the type of agent-environment interaction outlined in the previous section. The model can be used to reason about future states of a scenario process during planning, such that agents can reason about future states to optimize the long-term reward.

We present a POMDP model with factored states, as shown in the dynamic Bayesian network in Figure 4. Before explaining the Scenario-POMDP model in detail, we formalize it below.

**Definition 2.** (*Scenario-POMDP*). A *Scenario-POMDP* is a POMDP in which each state  $s \in S$  can be factored into a tuple  $s = (m, x, t)$ , where  $m$  is the domain-level state of the environment,  $x \in X$  is a scenario and  $t$  is a time index. A *Scenario-POMDP* has the following properties:

1. The scenario state variable  $x$  is partially observable, and state variables  $t$  and  $m$  are observable<sup>1</sup>. In state  $s = (m, x, t)$ , observation  $x_t$  is observed with probability 1, determined by  $x$  and  $t$ .
2. The transitions of  $m$  are determined by a predefined transition function. State variable  $t$  is always incremented by 1 after executing an action. The scenario state variable  $x$  is fixed.
3. The reward received in state  $s = (m, x, t)$  depends on  $x_t$ , defined by variable  $x$  and variable  $t$ , and depends on domain-level state  $m$  and action  $a$ .

A Scenario-POMDP models problems in which the interaction with the environment is represented by an MDP defining the domain-level actions and states, but the rewards also

<sup>1</sup>The Scenario-POMDP model can also be used if  $m$  is partially observable, which requires factored observations.

**input:** initial domain-level state  $m_0$ , horizon  $T$ , scenario set  $X$ , threshold  $\rho$

```

 $m \leftarrow m_0$ 
for  $t = 1, \dots, T$  do
   $o_t \leftarrow$  state of scenario process
   $o_{1,t} \leftarrow (o_1, \dots, o_t)$ 
   $w \leftarrow$  WEIGHTS( $o_{1,t}, X, \rho$ )
   $a \leftarrow$  POMCP( $m, w, t$ )
  execute action  $a$ 
   $m \leftarrow$  state obtained after executing  $a$  in state  $m$ 
end

```

### Algorithm 2: Scenario-POMCP.

depend on the state of the scenario process. Actions only affect the domain-level state  $m$ , and we assume that they do not influence the state transitions of the scenario process.

The state variable  $m$  represents the domain-level state of the environment, and the actual definition of  $m$  is dependent on the problem domain, as we will show later. The state variable  $x$  represents a scenario, where the scenario is a sequence of states as introduced in Definition 1. A scenario defines the state observations to be made in future timesteps, and therefore the scenario  $x$  is considered to be partially observable. A scenario cannot be fully observable, because this would imply that there is prior knowledge regarding future states of the scenario process. In this paper  $x$  denotes both a scenario and scenario state variable, but this makes the explanations more intuitive and clear. The state variable  $t$  is a variable representing the current timestep, and is fully observable. The Scenario-POMDP model can be considered as a MOMDP, which is a subclass of POMDPs for problems with mixed observability (Ong et al., 2010).

The observations in a Scenario-POMDP are exclusively determined by the factored state variables  $x$  and  $t$ . For scenario variable  $x$  and time variable  $t$ , the observation received by the agent is the observation at time  $t$  in scenario  $x$ . Suppose that the scenario  $x$  is (6, 9, 12) and the time state variable equals 2, then the agent will receive observation 9. This explains how the observations in Figure 4 depend on the scenario variable  $x$  and time variable  $t$ .

An action  $a$ , represented by the square in the figure, only affects the domain-level state  $m$  and does not influence the scenario  $x$ . For each action, the reward  $R$  is dependent on the domain-level state of the environment, as well as the observed state of the scenario process, defined by scenario state variable  $x$  and time variable  $t$ .

### 3.3 PLANNING FOR SCENARIO-POMDPs

We present a general planning algorithm, called Scenario-POMCP, for planning problems that can be formulated as a Scenario-POMDP. The description of the algorithm is

Table 1: Task Scheduling State Variables.

| VARIABLE                  | DESCRIPTION                               |
|---------------------------|---|
| $m_s^i$                   | state of task $i$ , for $i = 1, \dots, n$ |
| $m_a \in \{1, \dots, n\}$ | agent owning the token                    |
| $x \in X$                 | scenario                                  |
| $t \in \{1, \dots, T\}$   | time                                      |

shown in Algorithm 2. The initial domain-level state  $m_0$ , time horizon  $T$ , scenario set  $X$  and threshold  $\rho$  are given as input. The state observations  $o_t$  are used to define  $o_{1,t}$  and Algorithm 1 is used as a subroutine to assign weights to scenarios in  $X$ . The POMCP algorithm (Silver and Veness, 2010) is used as planning algorithm, which is an online planning algorithm for POMDPs based on Monte-Carlo tree search. This algorithm receives  $m$ ,  $w$  and  $t$  as input, and samples scenarios from  $X$  with a probability proportional to their weight in the vector  $w$ . We use POMCP because this algorithm can be adapted to sample scenarios based on weights, rather than sampling states from a belief state, and the algorithm is able to deal with a large number of states. The latter is relevant since the number of states of a Scenario-POMDP may grow very large. Eventually our algorithm executes the resulting action  $a$  before proceeding to the next timestep. At any timestep a new POMCP search tree is created, because there is no explicit link between the weights of consecutive timesteps. More implementation details are provided in the supplementary material.

## 4 SCENARIOS IN SMART GRIDS

In this section we formulate matching of demand with renewable supply in smart grids as a Scenario-POMDP, and we run simulations to compare the performance with other methods.

### 4.1 BACKGROUND

We consider  $n$  power demanding tasks, denoted by  $J = \{j_1, \dots, j_n\}$ , where each task  $j_i$  is parameterized by a duration  $l_i$ , release time  $r_i$ , deadline  $d_i$  and power demand  $p_i$ . Hence, we define each task  $j_i$  as a tuple  $j_i = (l_i, r_i, d_i, p_i)$ . A task is not allowed to start before its release time, must be finished by the deadline and cannot be preempted. The power demand  $p_i$  of task  $j_i$  represents the demand per timestep, which means that the total power consumption of task  $j_i$  equals  $l_i \cdot p_i$ .

There are two electricity sources: renewable energy derived from wind and conventional generation from the electricity grid. The available supply of conventional generation is assumed infinite and there is a cost function  $c(u)$  defining the

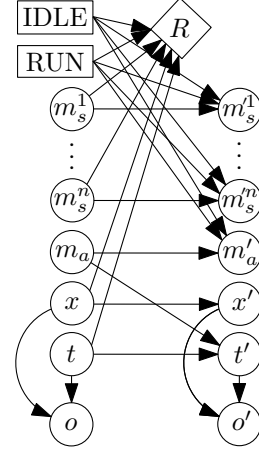


Figure 5: Dynamic Bayesian network of the Scenario-POMDP for task scheduling.

cost of consuming  $u$  units from the grid. We assume that renewable energy supply per timestep is finite, has zero cost and cannot be stored to be used in subsequent timesteps. The amount of renewable supply generated by wind is represented by a scenario  $x = (x_1, x_2, \dots, x_T)$  defining the number of units available at each timestep, where  $T$  is the time horizon.

A schedule  $S = (h_1, \dots, h_n)$  defines for each task  $j_i$  a starting time  $h_i$  satisfying the following conditions:

$$h_i \geq r_i, \quad h_i + l_i - 1 \leq d_i, \quad (i = 1, \dots, n).$$

The first condition states that task  $j_i$  cannot start before its release time  $r_i$  and the second condition defines that task  $j_i$  cannot run after the deadline  $d_i$ . The total demand  $D(S, t)$  of a schedule  $S = (h_1, \dots, h_n)$  at time  $t$  is defined as  $D(S, t) = \sum_{i=1}^n I_S(i, t) \cdot p_i$ , where  $I_S(i, t)$  is an indicator function that equals 1 if task  $j_i$  runs at time  $t$  in schedule  $S$ , and equals 0 otherwise. The number of required grid units  $U_{S,x}$  for schedule  $S$  and fixed scenario  $x = (x_1, x_2, \dots, x_T)$  can be computed as follows:

$$U_{S,x} = \sum_{t=1}^T \max(D(S, t) - x_t, 0).$$

The cost function  $c(U_{S,x})$  is used as an objective function to be minimized, in order to match demand and the renewable supply defined by the scenario.

### 4.2 SCENARIO-POMDP FORMULATION

We formulate the problem to start and defer tasks as a planning problem, in which we assume that an agent is associated with each task. A scenario  $x = (x_1, \dots, x_{24})$  is a sequence of wind state observations, where each  $x_i$  corresponds to the wind speed measured during hour  $i$ . The time

horizon  $T$  is equal to 24. Available renewable supply generated by wind is observed during the day, and the cost of running a task depends on the supply from wind and the decisions made for other tasks.

The problem is formulated as a Scenario-POMDP, using the factored state variables in Table 1. The state variables  $m_{s,1}, \dots, m_{s,n}$  define the states of the individual tasks, which encode properties such as release time, duration and deadline. The state variable  $m_a$  represents which agent is allowed to make a decision, which is used to reduce the size of the action space. More details are provided in the supplement. The factored state description is also visualized as a dynamic Bayesian network in Figure 5, which shows the correspondence with the Scenario-POMDP model in Figure 4.

An agent is able to execute two different actions: RUN and IDLE, corresponding to either running a task at a certain timeslot or doing nothing. The rewards are equal to the cost of running a task, multiplied by  $-1$ , which leads to a higher penalty if costly conventional generation is used. The observations automatically follow from the Scenario-POMDP model, as explained in Section 3.

### 4.3 EXPERIMENTS

In the experiments we run simulations to compare the proposed Scenario-POMDP formulation with other methods. We obtained historical hourly wind data from the Sotavento wind farm located in Galicia, Spain for 1708 days in the period from October 2008 until May 2013.<sup>2</sup> We consider the dataset as a long vector defining wind for 1708 consecutive days of 24 hours each, in which wind is measured in km/hr. For each subsequence of 24 hours, we define a scenario  $x = (x_1, \dots, x_{24})$ , which yields 40969 scenarios in total. In order to discretize the observation space, we round wind speed values to the nearest integer. The generated power  $Z(x, t)$  at time  $t$  in scenario  $x$  can be derived using a sigmoid power curve:

$$Z(x, t) = C \cdot (1 + e^{6 - \frac{2}{3}x_t})^{-1}$$

where  $C$  is a variable to define the capacity of the generator (Robu et al., 2012). For each task scheduling instance, we choose a scalar  $C$  such that  $Z(x, t) = \sum_{i=1}^n l_i \cdot p_i$ , which ensures that the total demand equals the available renewable supply.

We evaluate our planning algorithm on 200 task scheduling instances. Each instance consists of a set containing 6 tasks:  $J = \{j_1, j_2, \dots, j_6\}$ . We assign a duration between 3 and 7 to each task  $j_i$ , and a release time between 8 and 12, both sampled uniformly at random. The release times represent that tasks are released between 8AM and noon. The deadline  $d_i$  is set after 24 hours, to ensure that tasks

<sup>2</sup>Consult [www.sotaventogalicia.com](http://www.sotaventogalicia.com) for details.

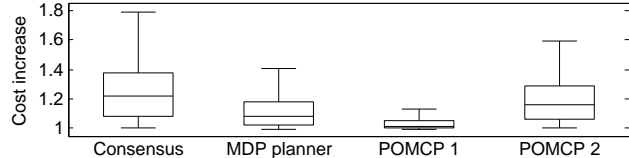


Figure 6: Performance comparison between planning with scenarios and other methods, without outliers.

Table 2: Statistics for Smart Grids Experiment 1.

|      | CONS. | MDP  | POMCP 1 | POMCP 2 |
|------|-------|------|---------|---------|
| Mean | 1.33  | 1.15 | 1.05    | 1.23    |
| Std  | 0.44  | 0.21 | 0.12    | 0.25    |
| Max  | 4.22  | 2.55 | 2.07    | 2.84    |

have finished by the end of the day. The power demand  $p_i$  equals 10 for each task, such that running tasks require 10 units at each timestep. The cost of consuming one unit from the grid is assumed to be 1. To define the renewable supply that is available during the day in each experiment, we sample an observation sequence from the scenario set. Days in which the renewable supply is relatively flat contain limited uncertainty. Therefore, we select scenarios where the renewable supply from time 1 to 6 and from time 13 to 18 is higher than the supply during the remaining hours. This guarantees that the renewable supply is unstable and varies during the day.

In our experiments we aim to compare the performance of Scenario-POMCP with the cost of optimal omniscient schedules. These assume that the supply throughout the day is known, which is a lower bound on the performance. We use mixed-integer programming with a 1 percent MIP gap to compute this for each instance. The scenario-based POMCP planner runs 200 iterations with an  $\epsilon$ -greedy exploration strategy, in which the probability to select random actions decreases linearly from 1 to 0, and threshold  $\rho$  in the weight computation equals 10. Additionally, we run an MDP planner that is based on Monte-Carlo tree search, and we also compare with a consensus task scheduling algorithm (Ströhle et al., 2014). More implementation details, and additional information regarding the consensus algorithm, are provided in the supplement.

The results of our comparison are shown in Figure 6, in which we show the performance relative to the cost of the optimal omniscient schedules. The cost of the optimal omniscient schedules is represented by 1, and the distributions show the performance relative to this cost. For example, if the cost is 1.2, this means that the cost is 20 percent higher than the cost of an optimal omniscient schedule. For readability reasons outliers have been omitted in the figure, and therefore additional statistics are provided in Table 2.

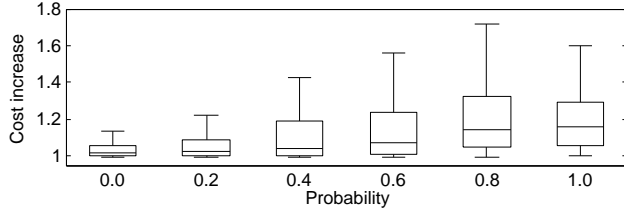


Figure 7: Cost increase for increasing probability to exclude the state observation sequence from  $X$ .

Table 3: Statistics for Smart Grids Experiment 2.

|      | 0.0  | 0.2  | 0.4  | 0.6  | 0.8  | 1.0  |
|------|------|------|------|------|------|------|
| Mean | 1.05 | 1.10 | 1.17 | 1.20 | 1.27 | 1.23 |
| Std  | 0.12 | 0.21 | 0.35 | 0.37 | 0.38 | 0.25 |
| Max  | 2.07 | 2.77 | 4.19 | 4.19 | 4.19 | 2.84 |

From the results we can conclude that the MDP planner performs slightly better than the consensus planner. For Scenario-POMCP we ran the algorithm with two different configurations. The column labeled POMCP 1 represents the case in which the observed state scenario is already present in the scenario set  $X$ , and then it performs much better than other methods. The column labeled POMCP 2 shows the results for the experiment in which the observed state scenario is never present in the set  $X$ . In those cases the performance is slightly worse, but still competitive with other methods.

In practice it can be expected that accurate scenario sets can be obtained from large historical datasets, and it is unlikely that the observed state scenario is never present in  $X$ . Therefore, we did an additional experiment in which the observed state scenario is excluded from  $X$  with a certain probability. This means that the algorithm sometimes encounters known scenarios, and in other cases the observed scenario is new. The results are shown in Figure 7 and Table 3, from which we can conclude that the performance of Scenario-POMCP is better if it is more likely that the observed state sequence is already part of the set. Therefore, better performance can be obtained by having a scenario set that covers all possible sequences of states accurately.

Our experiments make clear that the Scenario-POMDP model can be used for matching demand and uncertain supply in the smart grids domain. Additional results can be found in previous work (Walraven and Spaan, 2015).

## 5 SCENARIOS IN OPTION TRADING

In this section we show how scenarios can be used in financial stock markets.

### 5.1 BACKGROUND

A popular type of financial option is the European call option. This option gives the holder the right, but not the obligation, to buy a share at a prescribed point in time for a prescribed price regardless of the stock price. A European call option is parameterized by a strike price  $E$  and expiry date  $H$ , giving the holder the right to pay  $E$  for a share at time  $H$ . The value of the European call option at time  $H$  is  $\max(S(H) - E, 0)$ , where  $S(H)$  denotes the stock price at time  $H$ . If the strike price is lower than the stock price at expiry, the option holder can earn money by buying a share and selling it immediately on the market. If the strike price is higher, then the trader cannot gain anything. The value of a European call option can be determined using the Black-Scholes equation (Black and Scholes, 1973), for which a description is provided in the supplement. Reasoning about the future is necessary during trading, because if the stock price drops an option may become worthless.

### 5.2 SCENARIO-POMDP FORMULATION

We formulate buying and selling call options as a single-agent planning problem. A scenario  $x = (x_1, \dots, x_t)$  is defined as a sequence of stock price values, where  $x_i$  is the stock price observed at time  $i$ . We assume that there is an agent observing the market, and depending on the stock price it may decide to buy a call option, and if it owns a call option the agent may decide to sell the option to make profit. There is one type of call options the agent can buy. If the current stock price is  $j$ , the agent can buy a call option that expires after 10 days, with strike  $j$ .

The planning problem can be defined as a Scenario-POMDP, and we formulate the problem using the factored state variables in Table 4. The scenario variable  $x$  and time variable  $t$  are identical to the state variables  $x$  and  $t$  in a Scenario-POMDP, and they define the observations. The state variables  $m_o$ ,  $m_e$  and  $m_t$  represent the current state of the option portfolio of the agent. The variable  $m_o$  can be either true (T) or false (F), representing whether the agent currently owns a call option or not. If the agent owns a call option, the strike price of the option is represented by state variable  $m_e$ , and state variable  $m_t$  represents the number of days until expiry. The factored variables  $m_o$ ,  $m_e$  and  $m_t$  together define the state  $m$  of the option portfolio. The factored state description is also visualized as a dynamic Bayesian network in Figure 8, which shows the correspondence with the Scenario-POMDP model in Figure 4.

The agent can execute three different actions: BUY, SELL and NOOP. The action NOOP represents doing nothing, and the actions BUY and SELL correspond to buying and selling an option. The agent must sell once the option has expired. The rewards correspond to either paying a certain amount of money, or receiving a certain amount of money. The agent always pays the Black-Scholes value of the op-



Table 4: Option Trading State Variables.

| VARIABLE                  | DESCRIPTION                          |
|---------------------------|--------------------------------------|
| $m_o \in \{T, F\}$        | represents whether agent owns a call |
| $m_e \in \mathbb{N}$      | strike price of the call             |
| $m_t \in \{0, \dots, 9\}$ | time to expiry                       |
| $x \in X$                 | scenario                             |
| $t \in \{1, \dots, T\}$   | time                                 |

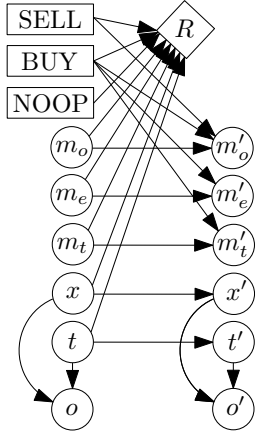


Figure 8: Dynamic Bayesian network of the Scenario-POMDP for option trading.

tion when buying an option, and receives the Black-Scholes value of the option when selling the option. The observations automatically follow from the Scenario-POMDP model, as explained in Section 3. A full description of the state transitions and rewards can be found in the supplement.

### 5.3 EXPERIMENTS

We did several experiments to evaluate the performance of the option trading agent on realistic data. To be able to reason about the stock price in the future, we obtained the historical daily close price values to build realistic scenario sets for shares in companies A and B. For company A, we use the data from January 2, 2001 to December 8, 2010, and for company B we use the data from September 27, 2000 to September 12, 2008. The stock price values are discretized, such that each price is a natural number. We enumerated subsequences of length 40 from the datasets to create a scenario set for each company.

To evaluate the performance, we simulate the stock market for each company for a period of at least 1000 days, with historical data that is more recent than the datasets we used to create the scenario set, which ensures that both datasets are distinct. For company A we simulate the stock

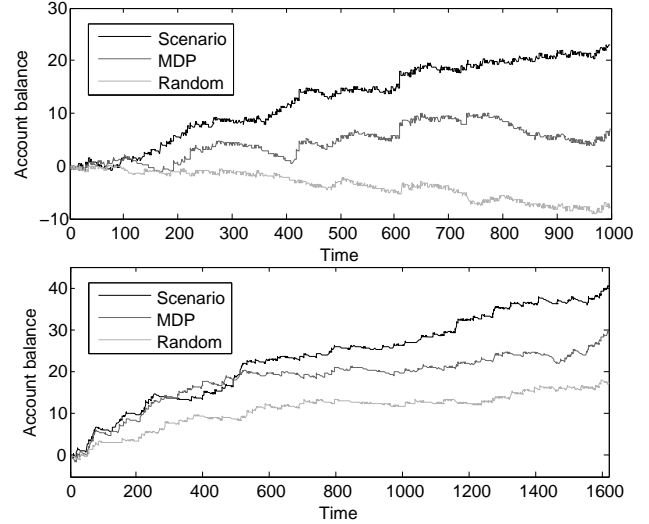


Figure 9: Account balance during simulations for company A (top) and company B (bottom).

price from December 9, 2010 to December 1, 2014. For company B we simulate the stock price from September 15, 2008 to February 20, 2015. We measured the average historical volatility using an Exponential Weighted Moving Average approach (Higham, 2008), and based on that we assume that the volatility of the market equals 0.4. The annual interest rate is assumed to be 0.03.

Our scenario-based agent uses Scenario-POMCP to select actions, which is implemented with a rolling time horizon. In our experiment the number of POMCP search iterations is set to 200, and the parameter  $\rho$  is equal to 10. POMCP uses UCB (Auer et al., 2002) as action selection heuristic with parameter 100 for company A and parameter 40 for company B.

In our experiments we aim to show that our method based on scenarios trades better than other methods. We can do this by keeping track of the account balance during the simulations, and we expect that an agent using the scenario planner earns more money than other methods. We also implemented an agent that is trading randomly, and an agent using an MDP formulation of the problem, which models the stock price as a Markov chain. The MDP-based agent uses Monte-Carlo tree search to select actions.

To compare the performance of the methods involved, we compare the trajectories defining the account balance during the simulations, as shown in Figure 9. For each method it shows the balance of the bank account for either 1000 or 1600 days. For each company, the scenario-based agent earns consistently more money in comparison to the other methods and, as expected, the random agent performs poorly. Based on our experiments we conclude that planning with scenarios has shown to perform well in this domain.

## 6 RELATED WORK

Ströhle et al. (2014) present a method to schedule tasks, where the uncertainty is also represented by weighted scenarios. Their method solves the problem for each scenario to reach consensus. A similarity with our work is that the method can be used to balance demand and supply, but our Scenario-POMDP formulation has been shown to outperform the consensus algorithm in case of high uncertainty. An important difference between the work by Ströhle et al. and this paper is that we define scheduling of tasks as a planning problem.

The problem to assign weights to expert opinions regarding the future is studied by Carvalho and Larson (2013). The method proposed is similar to our work because the similarity between opinions is measured by a distance function based on squared errors, which we also use to assign weights to scenarios, and our scenarios can also be considered as opinions regarding uncertain future events. In our work we always recompute weights associated with scenarios, whereas the work on opinion pools allows experts to update weights when new opinions become available.

Optimization using scenarios has been studied in the context of stochastic programming. Multi-stage stochastic programming problems can be considered as a subclass of Markov Decision Processes with a finite horizon (Defourny et al., 2011), and in this formalism the uncertainty is also represented by future scenarios, which is similar to our representation of state scenarios.

Exploiting factored structures in the POMCP algorithm has been studied by Amato and Oliehoek (2015). They propose a variant of POMCP that does not assume a factored model, but it uses factored value functions, which reduces the number of joint actions and joint histories in the multi-agent setting. A similarity is that we use a factored representation in each node of the POMCP search tree, which we can exploit to sample scenarios rather than states, but factored value functions have not been considered in our work.

## 7 CONCLUSIONS

In this paper we proposed a scenario-based approach to predict external factors that are difficult to model using a compact Markovian state. Scenarios represent sequences of states, and we have shown how a scenario can be weighted based on a sequence of states that occurred in the past, corresponding to the likelihood that a scenario perfectly predicts future states. In order to use the scenario representation in planning problems, we proposed a model called Scenario-POMDP, which enables agents to reason about future states during planning. To demonstrate the proposed model, we formulated matching of demand with renewable supply in smart grids as a Scenario-POMDP, and we have shown that our model can also be used to automati-

cally trade financial options. In both cases our Scenario-POMDP model performs better than other methods for decision making in these domains.

In future work we aim to study metrics for computing the distance between scenarios in which states are not represented by a single numerical value. Another direction for further research is defining a belief update for scenarios based on Bayes' rule, to replace the Monte-Carlo backups in our planner. Moreover, in our current work we assume that the domain-level state of the environment is observable, but we also want to study planning with scenarios for problems in which this part of the environment is partially observable, which requires factored observations.

### Acknowledgements

The work presented in this paper is funded by the Netherlands Organisation for Scientific Research (NWO), as part of the Uncertainty Reduction in Smart Energy Systems program. We would like to thank Mathijs de Weerd for pointing out the dataset of the Sotavento wind farm.

### References

- Amato, C. and Oliehoek, F. A. (2015). Scalable Planning and Learning for Multiagent POMDPs. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 1995–2002.
- Auer, P., Cesa-Bianchi, N., and Fischer, P. (2002). Finite-time Analysis of the Multiarmed Bandit Problem. In: *Machine Learning* 47.2-3, pp. 235–256.
- Black, F. and Scholes, M. (1973). The Pricing of Options and Corporate Liabilities. In: *The Journal of Political Economy* 81.3, pp. 637–654.
- Carvalho, A. and Larson, K. (2013). A Consensual Linear Opinion Pool. In: *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 2518–2524.
- Defourny, B., Ernst, D., and Wehenkel, L. (2011). Multi-stage Stochastic Programming: A Scenario Tree Based Approach to Planning under Uncertainty. In: *Decision Theory Models for Applications in Artificial Intelligence: Concepts and Solutions*. Ed. by L. Sucar, E. Morales, and J. Hoey. Information Science Publishing.
- Giebel, G., Brownsword, R., Kariniotakis, G., Denhard, M., and Draxl, C. (2011). *The State-Of-The-Art in Short-Term Prediction of Wind Power*. Deliverable of the European research project ANEMOS.plus.
- Hassan, M. R. and Nath, B. (2005). Stock Market Forecasting Using Hidden Markov Model: A New Approach. In: *Proceedings of the 5th International Conference on Intelligent Systems Design and Applications*, pp. 192–196.
- Higham, D. J. (2008). *An Introduction to Financial Option Valuation*. Cambridge University Press.

- Kaelbling, L. P., Littman, M. L., and Cassandra, A. R. (1998). Planning and acting in partially observable stochastic domains. In: *Artificial Intelligence* 101.1, pp. 99–134.
- Moura, P. S. and De Almeida, A. T. (2010). The role of demand-side management in the grid integration of wind power. In: *Applied Energy* 87.8, pp. 2581–2588.
- Ong, S. C. W., Png, S. W., Hsu, D., and Lee, W. S. (2010). Planning under Uncertainty for Robotic Tasks with Mixed Observability. In: *The International Journal of Robotics Research* 29.8, pp. 1053–1068.
- Papadimitriou, C. H. and Tsitsiklis, J. N. (1987). The complexity of Markov decision processes. In: *Mathematics of Operations Research* 12.3, pp. 441–450.
- Pineau, J., Gordon, G., and Thrun, S. (2003). Point-based value iteration: An anytime algorithm for POMDPs. In: *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 1025–1030.
- Puterman, M. L. (1994). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc.
- Robu, V., Kota, R., Chalkiadakis, G., Rogers, A., and Jennings, N. R. (2012). Cooperative Virtual Power Plant Formation Using Scoring Rules. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 370–376.
- Silver, D. and Veness, J. (2010). Monte-Carlo Planning in Large POMDPs. In: *Advances in Neural Information Processing Systems*, pp. 2164–2172.
- Spaan, M. T. J. and Vlassis, N. (2005). Perseus: Randomized Point-based Value Iteration for POMDPs. In: *Journal of Artificial Intelligence Research* 24, pp. 195–220.
- Ströhle, P., Gerding, E. H., De Weerd, M. M., Stein, S., and Robu, V. (2014). Online Mechanism Design for Scheduling Non-Preemptive Jobs under Uncertain Supply and Demand. In: *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*, pp. 437–444.
- Walraven, E. and Spaan, M. T. J. (2015). A Scenario State Representation for Scheduling Deferrable Loads under Wind Uncertainty. In: *The 10th Annual Workshop on Multiagent Sequential Decision Making under Uncertainty*.
- Witwicki, S., Melo, F. S., Capitán, J., and Spaan, M. T. J. (2013). A Flexible Approach to Modeling Unpredictable Events in MDPs. In: *Proceedings of the International Conference on Automated Planning and Scheduling*, pp. 260–268.

---

# Generalization Bounds for Transfer Learning under Model Shift

---

**Xuezhi Wang**  
Computer Science Dept.  
Carnegie Mellon University  
Pittsburgh, PA 15213

**Jeff Schneider**  
Robotics Institute  
Carnegie Mellon University  
Pittsburgh, PA 15213

## Abstract

Transfer learning (sometimes also referred to as domain-adaptation) algorithms are often used when one tries to apply a model learned from a fully labeled source domain, to an unlabeled target domain, that is similar but not identical to the source. Previous work on covariate shift focuses on matching the marginal distributions on observations  $X$  across domains while assuming the conditional distribution  $P(Y|X)$  stays the same. Relevant theory focusing on covariate shift has also been developed. Recent work on transfer learning under model shift deals with different conditional distributions  $P(Y|X)$  across domains with a few target labels, while assuming the changes are smooth. However, no analysis has been provided to say when these algorithms work. In this paper, we analyze transfer learning algorithms under the model shift assumption. Our analysis shows that when the conditional distribution changes, we are able to obtain a generalization error bound of  $O(\frac{1}{\lambda_*\sqrt{n_t}})$  with respect to the labeled target sample size  $n_t$ , modified by the smoothness of the change ( $\lambda_*$ ) across domains. Our analysis also sheds light on conditions when transfer learning works better than no-transfer learning (learning by labeled target data only). Furthermore, we extend the transfer learning algorithm from a single source to multiple sources.

## 1 INTRODUCTION

In a classical transfer learning setting (see Fig. 1), we have a source domain with sufficient fully labeled data, and a target domain with data that has little or no labels. These two domains are related but not identical, and the usual assumption is that there is some knowledge that can be transferred from the source domain to the target domain. Examples of transfer learning applied in the real-world include, adapting

classification models for different products, and transferring across diseases on medical data (Pan et al. (2009)). A number of different transfer learning techniques have been introduced in the past, e.g., algorithms dealing with covariate shift (Shimodaira (2000), Huang et al. (2007), Gretton et al. (2007)). Related theoretical analyses on covariate shift have also been developed, e.g., for sample size  $m$  in the source domain and sample size  $n$  in the target domain, the analysis of Mansour et al. (2009) achieves a rate of  $O(m^{-1/2} + n^{-1/2})$ , and convergence of reweighted means in feature space achieves rate  $O((1/m + 1/n)^{1/2})$  (Huang et al. (2007)).

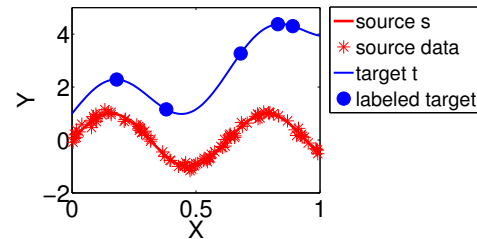


Figure 1: Transfer learning example:  $m$  source data points  $\{X^s, Y^s\}$  (red),  $n$  target data points  $\{X^t, Y^t\}$  (blue), and  $n_t$  labeled target points (solid blue circles). Here  $X$  denotes the input features and  $Y$  denotes the output labels.

However, not much work on transfer learning has considered the case when a few labels in the target domain are available. Also little work has been done when conditional distributions are allowed to change (defined as **model shift**). Recently, algorithms dealing with transfer learning under model shift have been proposed, where the changes on conditional distributions are assumed to be smooth (Wang et al. (2014)). However, no theoretical analysis has been provided for these approaches.

In this paper, we develop theoretical analysis for transfer learning algorithms under the model shift assumption. Our analysis shows that even when the conditional distributions are allowed to change across domains, we are still able to obtain a generalization bound of  $O(\frac{1}{\lambda_*\sqrt{n_t}})$  with respect to

the labeled target sample size  $n_t$ , modified by the smoothness of the transformation parameters ( $\lambda_*$ ) across domains. Our analysis also sheds light on conditions when transfer learning works better than no-transfer learning. We show that under certain smoothness assumptions it is possible to obtain a favorable convergence rate with transfer learning compared to no transfer at all. Furthermore, using the generalization bounds we derived in this paper, we are able to extend the transfer learning algorithm from a single source to multiple sources, where each source is assigned a weight that indicates how helpful it is for transferring to the target.

We illustrate our theoretical results by empirical comparisons on both synthetic data and real-world data. Our results demonstrate cases where we obtain the same rate as no-transfer learning, and cases where we obtain a favorable rate with transfer learning under certain smoothness assumptions, which coincide with our theoretical analysis. In addition, experiments on the real data show that our algorithm for reweighting multiple sources yields better results than existing state-of-the-art algorithms.

## 2 RELATED WORK

Traditional methods for transfer learning use relatively restrictive assumptions, where specific parts of the learning model are assumed to be carried over between tasks. For example, Mihalkova et al. (2007) transfers relational knowledge across domains using Markov logic networks. Niculescu-Mizil & Caruana (2007) learns Bayes Net structures by biasing learning toward similar structures for each task. Do & Ng (2005) and Raina et al. (2006) assume that models for related tasks share same parameters or prior distributions of hyperparameters.

A large part of transfer learning work is devoted to the problem of covariate shift (Shimodaira (2000), Huang et al. (2007), Gretton et al. (2007)), where the assumption is that only the marginal distribution  $P(X)$  differs across domains but the conditional distribution  $P(Y|X)$  stays the same. The kernel mean matching (KMM) method (Huang et al. (2007), Gretton et al. (2007)), is one of the algorithms that deal with covariate shift. Huang et al. (2007) proved the convergence of reweighted means in the feature space, and showed that their method results in almost unbiased risk estimates. More recent research (Zhang et al. (2013)) focused on modeling target shift ( $P(Y)$  changes), conditional shift ( $P(X|Y)$  changes), and a combination of both. The assumption for target shift is that  $X$  depends causally on  $Y$ , thus  $P(Y)$  can be re-weighted to match the distributions on  $X$  across domains. The authors also provided some theoretical analysis of the conditions when  $P(X|Y)$  is identifiable. Both covariate shift and target/conditional shift make no use of target labels  $Y^t$ , even if some are available. For transfer learning under model shift, there could be a difference in  $P(Y|X)$  that can not simply be captured by

the differences in  $P(X)$ , hence neither covariate shift nor target/conditional shift will work well under the model shift assumption.

A number of theoretical analyses on domain adaptation have also been developed. Ben-David et al. (2006) presented VC-dimension-based generalization bounds for adaptation in classification tasks. Later Blitzer et al. (2007) extended the work with a bound on the error rate under a weighted combination of the source data. Mansour et al. (2009) introduced a discrepancy distance suitable for arbitrary loss functions and derived new generalization bounds for domain adaptation for a wide family of loss functions. However, most of the work mentioned above deals with domain adaptation under the covariate shift assumption, which means they still assume the conditional distribution stays the same across domains, or the labeling functions in the two domains share strong proximity in order for adaptation to be possible. For example, one of the bounds derived in Mansour et al. (2009) has a term  $\mathcal{L}(h_Q^*, h_P^*)$  related to the average loss between the minimizer  $h_Q^*$  in the source domain and the minimizer  $h_P^*$  in the target domain, which could be fairly large when there exists a constant offset between the two labeling functions.

In Wang et al. (2014), the authors proposed a transfer learning algorithm to handle the general case where  $P(Y|X)$  changes smoothly across domains. However, the authors fail to make explicit connections between the smoothness assumption and the generalization bounds for transfer learning. They do not show whether the performance will degrade when the smoothness assumption is relaxed, and whether the smoothness assumption yields a lower generalization error for transfer learning than no-transfer learning.

Similarly, most work in transfer learning with multiple sources focuses only on  $P(X)$ . For example, Mansour et al. (2008) proposed a distribution weighted combining rule of source hypotheses using the input distribution  $P(X)$  for both source and target. This approach requires estimating the distribution  $D_i(x)$  of source  $i$  on a target point  $x$  from large amounts of unlabeled points typically available from the source, which might be difficult in real applications with high-dimensional features. Other existing work focuses on finding the set of sources that are closely related to the target (Crammer et al. (2008)), or a reweighting of sources based on prediction errors (Yao and Doretto, (2010)). Chattopadhyay et al. (2011) proposed a conditional probability based weighting scheme under a joint optimization framework, which leads to a reweighting of sources that prefers more consistent predictions on the target. However, these existing approaches do not consider the problem that there might exist shifts in the conditional distribution from source to the target, and how the smoothness of this shift can help in learning the target, which is the main issue addressed in this paper.

### 3 TRANSFER LEARNING UNDER MODEL SHIFT: A REVIEW OF THE ALGORITHMS

**Notation:** Let  $\mathcal{X} \in \mathcal{R}^d$  and  $\mathcal{Y} \in \mathcal{R}$  be the input and output space for both the source and the target domain. We are given a set of  $m$  labeled data points,  $(x_i^s, y_i^s) \in (X^s, Y^s)$ ,  $i = 1, \dots, m$ , from the source domain. We are also given a set of  $n$  target data points,  $X^t$ , from the target domain. Among these we have  $n_l$  labeled target data points, denoted as  $(X^{tL}, Y^{tL})$ . The unlabeled part of  $X^t$  is denoted as  $X^{tU}$ , with unknown labels  $Y^{tU}$ . For simplicity let  $z \in \mathcal{Z} = \mathcal{X} \times \mathcal{Y}$  denote the pair of  $(x, y)$ , and we use  $z^s, z^t, z^{tL}$  for the source, target, and labeled target, correspondingly. We assume  $X^s, X^t$  are drawn from the same  $P(X)$  throughout the paper since we focus more on  $P(Y|X)$ <sup>1</sup>. If necessary  $P(X)$  can be easily matched by various methods dealing with covariate shift (e.g. Kernel Mean Matching) without the use of  $Y$ .

Let  $\mathcal{H}$  be a reproducing kernel Hilbert space with kernel  $K$  such that  $K(x, x) \leq \kappa^2 < \infty$  for all  $x \in X$ . Let  $\|\cdot\|_k$  denote the corresponding RKHS norm. Let  $\phi$  denote the feature mapping on  $x$  associated with kernel  $K$ , and  $\Phi(X)$  denote the matrix where the  $i$ -th column is  $\phi(x_i)$ . Denote  $K_{XX'}$  as the kernel computed between matrix  $X$  and  $X'$ , i.e.,  $K_{ij} = k(x_i, x'_j)$ . When necessary, we use  $\psi$  to denote the feature map on  $y$ , and the corresponding matrix as  $\Psi(Y)$ . For a hypothesis  $h \in \mathcal{H}$ , assume that  $|h(x)| \leq M$  for some  $M > 0$ . Also assume bounded label set  $|y| \leq M$ . We use  $\ell_2$  loss as the loss function  $l(h(x), y)$  throughout this paper, which is  $\sigma$ -admissible, i.e.,

$$\forall x, y, \forall h, h', |l(h(x), y) - l(h'(x), y)| \leq \sigma |h(x) - h'(x)|. \quad (1)$$

It is easy to see that  $\sigma = 4M$  for bounded  $h(x)$  and  $y$ . Note the loss function is also bounded,  $l(h(x), y) \leq 4M^2$ .

Next we will briefly review two algorithms introduced in Wang et al. (2014) that handle transfer learning under model shift: the first is conditional distribution matching, and the second is two-stage offset estimation.

#### (1) Conditional Distribution Matching (CDM).

The basic idea of CDM is to match the conditional distributions  $P(Y|X)$  for the source and the target domain. Since there is a difference in  $P(Y|X)$  across domains, these two conditional distributions cannot be matched directly. Therefore, the authors propose to make a parameterized-location-scale transform on the source labels  $Y^s$ :

$$Y^{new} = Y^s \odot \mathbf{w}(X^s) + \mathbf{b}(X^s),$$

where  $\mathbf{w}$  denotes the scale transform,  $\mathbf{b}$  denotes the location transform, and  $\odot$  denotes the Hadamard (elementwise)

<sup>1</sup>This assumption is only required in our analysis for simplicity. It can be relaxed when applying the algorithms.

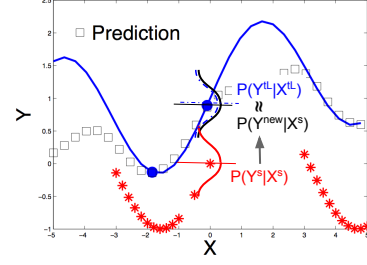


Figure 2: Illustration of the conditional distribution matching algorithm: red (source), blue (target).

product.  $\mathbf{w}$  and  $\mathbf{b}$  are non-linear functions of  $X$  which allows a non-linear transform from  $Y^s$  to  $Y^{new}$ .

The objective is to use the transformed conditional distribution in the source domain  $P(Y^{new}|X^s)$ , to match the conditional distribution in the target domain,  $P(Y^{tL}|X^{tL})$ , such that the transformation parameter  $\mathbf{w}$  and  $\mathbf{b}$  can be learned through optimization. The matching on  $P(Y|X)$  is achieved by minimizing the discrepancy of the mean embedding of  $P(Y|X)$  with a regularization term:

$$\min_{\mathbf{w}, \mathbf{b}} L + L_{reg}, \text{ where} \quad (2)$$

$$L = \|\hat{\mathcal{U}}[P_{Y^{new}|X^s}] - \hat{\mathcal{U}}[P_{Y^{tL}|X^{tL}}]\|_k^2,$$

$$L_{reg} = \lambda_{reg} (\|\mathbf{w} - \mathbf{1}\|^2 + \|\mathbf{b}\|^2),$$

where  $\mathcal{U}[P_{Y|X}]$  is the mean embedding of the conditional distribution  $P(Y|X)$  (Song et al. (2009)), and  $\hat{\mathcal{U}}[P_{Y|X}]$  is the empirical estimation of  $\mathcal{U}[P_{Y|X}]$  based on samples  $X, Y$ . Further the authors make a smoothness assumption on the transformation, i.e.,  $\mathbf{w}, \mathbf{b}$  are parameterized using:  $\mathbf{w} = R\mathbf{g}, \mathbf{b} = R\mathbf{h}$ , where  $R = K_{X^s X^s} (K_{X^s X^s} + \lambda_R I)^{-1}$ , and  $\mathbf{g}, \mathbf{h} \in \mathbb{R}^{m \times 1}$  are the new parameters to optimize in the objective. After obtaining  $\mathbf{g}, \mathbf{h}$  (or equivalently  $\mathbf{w}, \mathbf{b}$ ),  $Y^{new}$  is computed based on the transformation. Finally the prediction on  $X^{tU}$  is based on the merged data:  $(X^s, Y^{new}) \cup (X^{tL}, Y^{tL})$ .

Fig 2 shows an illustration of the conditional distribution matching algorithm. As we can see from the figure,  $Y^s$  is transformed to  $Y^{new}$  such that  $P(Y^{new}|X^s)$  and  $P(Y^{tL}|X^{tL})$  can be approximately matched together.

**Remark.** Here we analyze what happens when the smoothness assumption is relaxed. It is easy to derive that, when setting  $\mathbf{w} = \mathbf{1}, \mathbf{b} = \mathbf{0}$ , we can directly solve for  $Y^{new}$  by taking the derivative of  $L$  with respect to  $Y^{new}$ , and we get:

$$K_{X^s X^s} (K_{X^s X^s} + \lambda I)^{-1} Y^{new} = K_{X^s X^{tL}} (K_{X^{tL} X^{tL}} + \lambda I)^{-1} Y^{tL}, \quad (3)$$

where  $\lambda$  is some regularization parameter to make sure the kernel matrix is invertible. In other words, the smoothed  $Y^{new}$  is given by the prediction on the source using only labeled target data. Hence  $Y^{new}$  provides no extra information for prediction on the target, compared with using the labeled target data alone.

## (2) Two-stage Offset Estimation (Offset).

The idea of Offset is to model the target function  $f^t$  using the source function  $f^s$  and an offset,  $f^o = f^t - f^s$ , while assuming that the offset function is smoother than the target function. Specifically, using kernel ridge regression (KRR) to estimate all three functions, the algorithm works as follows:

- (1) Model the source function using the source data, i.e.,  $f^s(x) = K_{xX^s}(K_{X^sX^s} + \lambda I)^{-1}Y^s$ .
- (2) Model the offset function by the difference between the true target labels and the predicted target labels, i.e.,  $f^o(X^{tL}) = Y^{tL} - f^s(X^{tL})$ .
- (3) Transform  $Y^s$  to  $Y^{new}$  by adding the offset, i.e.,  $Y^{new} = Y^s + f^o(X^s)$ , where  $f^o(X^s) = K_{X^sX^{tL}}(K_{X^{tL}X^{tL}} + \lambda I)^{-1}f^o(X^{tL})$ .
- (4) Train a model on  $\{X^s, Y^{new}\} \cup \{X^{tL}, Y^{tL}\}$ , and use the model to make predictions on  $X^{tU}$ .

We would like to answer: under what conditions these transfer learning algorithms will work better than no-transfer learning, and how the smoothness assumption affects the generalization bounds for these algorithms.

## 4 ANALYSIS OF CONDITIONAL DISTRIBUTION MATCHING

In this section, we analyze the generalization bound for the conditional distribution matching (CDM) approach.

### 4.1 RISK ESTIMATES FOR CDM

We use stability analysis on the algorithm to estimate the generalization error. First we have:

**Theorem 1.** (Bousquet & Elisseeff (2002), Theorem 12 and Example 3) Consider a training set  $S = \{z_1 = (x_1, y_1), \dots, z_m = (x_m, y_m)\}$  drawn i.i.d. from an unknown distribution  $D$ . Let  $l$  be the  $\ell_2$  loss function which is  $\sigma$ -admissible with respect to  $\mathcal{H}$ , and  $l \leq 4M^2$ . The Kernel Ridge Regression algorithm defined by:

$$A_S = \arg \min_{h \in \mathcal{H}} \frac{1}{m} \sum_{i=1}^m l(h, z_i) + \lambda \|h\|_k^2$$

has uniform stability  $\beta$  with respect to  $l$  with  $\beta \leq \frac{\sigma^2 \kappa^2}{2\lambda m}$ .

In addition, let  $R = \mathbb{E}_z[l(A_S, z)]$  be the generalization error, and  $R_{emp} = \frac{1}{m} \sum_{i=1}^m l(A_S, z_i)$  be the empirical error, then the following holds with probability at least  $1 - \delta$ ,

$$R \leq R_{emp} + \frac{\sigma^2 \kappa^2}{\lambda m} + \left(\frac{2\sigma^2 \kappa^2}{\lambda} + 4M^2\right) \sqrt{\frac{\ln(1/\delta)}{2m}}.$$

In CDM, the prediction on the unlabeled target data points is given by merging the transformed source data and the labeled target data, i.e.,  $(X^s, Y^{new}) \cup (X^{tL}, Y^{tL})$ . Hence

we need to bound the difference between the empirical error on the merged data and the generalization error (risk) in the target domain.

Denote  $\tilde{z}_i = (\tilde{x}_i, \tilde{y}_i) \in (\tilde{X}, \tilde{Y})$ , where  $\tilde{X}, \tilde{Y}$  represents the merged data:  $\tilde{X} = X^s \cup X^{tL}, \tilde{Y} = Y^{new} \cup Y^{tL}$ . Let  $h^* \in \mathcal{H}$  be the minimizer on the merged data, i.e.,

$$h^* = \arg \min_{h \in \mathcal{H}} \frac{1}{m + n_l} \sum_{i=1}^{m+n_l} l(h, \tilde{z}_i) + \lambda \|h\|_k^2.$$

Then the following theorem holds:

**Theorem 2.** Assume the conditions in Theorem 1 hold. Also assume  $\|\hat{\mathcal{U}}[P_{Y^{new}|X^s}] - \hat{\mathcal{U}}[P_{Y^{tL}|X^{tL}}]\|_k \leq \epsilon$  after we optimize objective Eq. 2. The following holds with probability at least  $1 - \delta$ :

$$\begin{aligned} & \left| \frac{1}{m + n_l} \sum_{i=1}^{m+n_l} l(h^*, \tilde{z}_i) - E_{z^t}[l(h^*, z^t)] \right| \\ & \leq 4M(\epsilon \kappa + C(\lambda_c^{1/2} + (n_l \lambda_c)^{-1/2})) + \\ & \quad \frac{\sigma^2 \kappa^2}{\lambda_t(m + n_l)} + \left(\frac{2\sigma^2 \kappa^2}{\lambda_t} + 4M^2\right) \sqrt{\frac{\ln(1/\delta)}{2(m + n_l)}}, \end{aligned}$$

where  $\lambda_c$  is the regularization parameter used in estimating  $\hat{\mathcal{U}}[P_{Y^{tL}|X^{tL}}] = \Psi(Y^{tL})(K_{X^{tL}X^{tL}} + \lambda_c n_l I)^{-1} \Phi^\top(X^{tL})$ , and  $\lambda_t$  is the regularization parameter when estimating the target function.  $C > 0$  is some constant.

*Proof.* Let  $\bar{z}_i = (\bar{x}_i, \bar{y}_i) \in (\bar{X}, \bar{Y})$ , where  $\bar{X}, \bar{Y}$  are the auxiliary samples with  $\bar{X} = X^s \cup X^{tL}, \bar{Y} = \bar{Y}_s^t \cup Y^{tL}$ , where  $\bar{Y}_s^t$  are pseudo labels in the target domain for the source data points  $X^s$ . Using triangle inequality we can decompose the LHS by:

$$\begin{aligned} & \left| \frac{1}{m + n_l} \sum_{i=1}^{m+n_l} l(h^*, \tilde{z}_i) - E_{z^t}[l(h^*, z^t)] \right| \\ & \leq \left| \frac{1}{m + n_l} \sum_{i=1}^{m+n_l} l(h^*, \tilde{z}_i) - \frac{1}{m + n_l} \sum_{i=1}^{m+n_l} l(h^*, \bar{z}_i) \right| \\ & \quad + \left| \frac{1}{m + n_l} \sum_{i=1}^{m+n_l} l(h^*, \bar{z}_i) - E_{z^t}[l(h^*, z^t)] \right| \end{aligned}$$

The second term is easy to bound since it is simply the difference between the empirical error and the generalization error in the target domain with effective sample size  $n_l + m$ , thus using Theorem 1, we have

$$\begin{aligned} & \left| \frac{1}{m + n_l} \sum_{i=1}^{m+n_l} l(h^*, \bar{z}_i) - E_{z^t}[l(h^*, z^t)] \right| \\ & \leq \frac{\sigma^2 \kappa^2}{\lambda_t(m + n_l)} + \left(\frac{2\sigma^2 \kappa^2}{\lambda_t} + 4M^2\right) \sqrt{\frac{\ln(1/\delta)}{2(m + n_l)}}. \end{aligned} \quad (4)$$

To bound the first term, we have

$$\begin{aligned}
& \left| \frac{1}{m+n_l} \sum_{i=1}^{m+n_l} l(h^*, \tilde{z}_i) - \frac{1}{m+n_l} \sum_{i=1}^{m+n_l} l(h^*, \bar{z}_i) \right| \\
& \leq \frac{1}{m+n_l} \sum_{i=1}^{m+n_l} |l(h^*, \tilde{z}_i) - l(h^*, \bar{z}_i)| \\
& \leq \frac{1}{m+n_l} \sum_{i=1}^m 4M |y_i^{new} - \mathcal{U}[P_{Y^t|X^t}] \phi(x_i^s)| \\
& \leq \frac{4M}{m+n_l} \sum_{i=1}^m (|\hat{\mathcal{U}}[P_{Y^{new}|X^s}] \phi(x_i^s) - \hat{\mathcal{U}}[P_{Y^{tL}|X^{tL}}] \phi(x_i^s)| \\
& \quad + |\hat{\mathcal{U}}[P_{Y^{tL}|X^{tL}}] \phi(x_i^s) - \mathcal{U}[P_{Y^t|X^t}] \phi(x_i^s)|) \\
& \leq \frac{4M}{m+n_l} \sum_{i=1}^m (|\hat{\mathcal{U}}[P_{Y^{new}|X^s}] - \hat{\mathcal{U}}[P_{Y^{tL}|X^{tL}}]|_k \sqrt{k(x, x)} \\
& \quad + |\hat{\mathcal{U}}[P_{Y^{tL}|X^{tL}}] \phi(x_i^s) - \mathcal{U}[P_{Y^t|X^t}] \phi(x_i^s)|) \\
& \leq 4M(\epsilon\kappa + C(\lambda_c^{1/2} + (n_l\lambda_c)^{-1/2})), \tag{5}
\end{aligned}$$

where in the last inequality, the second term is bounded using Theorem 6, Song et al. (2009).

Now combining Eq. 5 and Eq. 4 concludes the proof.  $\square$

## 4.2 TIGHTER BOUNDS UNDER SMOOTH PARAMETERIZATION

Theorem 2 suggests that using CDM, the empirical risk converges to the expected risk at a rate of

$$O(\lambda_c^{1/2} + (n_l\lambda_c)^{-1/2} + \lambda_t^{-1}(m+n_l)^{-1/2}). \tag{6}$$

In the following, we show how the smoothness parameterization in CDM helps us obtain faster convergence rates.

Under the smoothness assumption on the transformation,  $\mathbf{w}$ ,  $\mathbf{b}$  are parameterized using:  $\mathbf{w} = R\mathbf{g}$ ,  $\mathbf{b} = R\mathbf{h}$ , where  $R = K_{X^s X^s}(K_{X^s X^s} + \lambda_R I)^{-1}$ . For simplicity we assume the same  $\lambda_R$  for both  $\mathbf{w}$  and  $\mathbf{b}$ . Similar to the derivation in Eq. 5, we have

$$\begin{aligned}
& |y_i^{new} - \mathcal{U}[P_{Y^t|X^t}] \phi(x_i^s)| \\
& = |\hat{\mathcal{U}}[P_{Y^{new}|X^s}] \phi(x_i^s) - \hat{\mathcal{U}}[P_{Y^{tL}|X^{tL}}] \phi(x_i^s)| \\
& \quad + |\hat{\mathcal{U}}[P_{Y^{tL}|X^{tL}}] \phi(x_i^s) - \mathcal{U}[P_{Y^t|X^t}] \phi(x_i^s)| \\
& \leq \epsilon\kappa + |\hat{\mathcal{U}}[P_{w^{tL}|X^{tL}}] \phi(x_i^s) - \mathcal{U}[P_{w^t|X^t}] \phi(x_i^s)| \cdot |y_i^s| \\
& \quad + |\hat{\mathcal{U}}[P_{b^{tL}|X^{tL}}] \phi(x_i^s) - \mathcal{U}[P_{b^t|X^t}] \phi(x_i^s)| \\
& \leq \epsilon\kappa + C_1(\lambda_R^{1/2} + (n_l\lambda_R)^{-1/2})M + C_2(\lambda_R^{1/2} + (n_l\lambda_R)^{-1/2})|y_i^s| \\
& \leq \epsilon\kappa + C'(\lambda_R^{1/2} + (n_l\lambda_R)^{-1/2}). \tag{7}
\end{aligned}$$

Hence we can update the bound in Eq. 5 by:

$$\begin{aligned}
& \left| \frac{1}{m+n_l} \sum_{i=1}^{m+n_l} l(h^*, \tilde{z}_i) - \frac{1}{m+n_l} \sum_{i=1}^{m+n_l} l(h^*, \bar{z}_i) \right| \\
& \leq 4M(\epsilon\kappa + C'(\lambda_R^{1/2} + (n_l\lambda_R)^{-1/2})). \tag{8}
\end{aligned}$$

It is easy to see that Eq. 4 remains the same. Hence, the rate for CDM under the smooth parametrization is:

$$O(\lambda_R^{1/2} + (n_l\lambda_R)^{-1/2} + \lambda_t^{-1}(m+n_l)^{-1/2}). \tag{9}$$

In transfer learning we usually assume the number of source data is sufficient, i.e.,  $m \rightarrow \infty$ . Comparing Eq. 9 with Eq. 6 we can see that, when the number of labeled points  $n_l$  is small, the term  $(n_l\lambda_c)^{-1/2}$  in Eq. 6 and the term  $(n_l\lambda_R)^{-1/2}$  in Eq. 9 take over. If we further assume that the transformation  $\mathbf{w}$  and  $\mathbf{b}$  are smoother functions with respect to  $X$  than the target function with respect to  $X$ , i.e.,  $\lambda_R > \lambda_c$ , then Eq. 9 is more favorable. On the other hand, when the number of labeled target points  $n_l$  is large enough for the first term  $\lambda_c^{1/2}$  in Eq. 6 and the first term  $\lambda_R^{1/2}$  in Eq. 9 to take over, then it is reasonable to use a  $\lambda_R$  closer to  $\lambda_c$  to get a similar convergence rate as in Eq. 6. Intuitively, when the number of labeled target points is large enough, it is not very helpful to transfer from the source for target prediction.

**Remark.** Note that in Eq. 6 and Eq. 9, an ideal choice of  $\lambda$  close to  $1/\sqrt{n_l}$  can minimize  $\lambda^{1/2} + (n_l\lambda)^{-1/2}$ . However, note that the generalization bound is the difference between the expected risk  $R$  and the empirical risk  $R_{emp}$ , and a  $\lambda$  that minimizes the generalization bound does not necessarily minimize the expected risk  $R$ , since the empirical risk  $R_{emp}$  (which is also affected by  $\lambda$ ) can still be large. To obtain a relatively small empirical risk,  $\lambda$  should be determined by the smoothness of the offset/target function, since it is the regularization parameter when estimating the offset/target. In practice  $\lambda$  is chosen by cross validation on the labeled data, and is not necessarily close to  $1/\sqrt{n_l}$ . For example, on real data we find that  $\lambda$  is usually chosen to be in the range of  $1e-2$  to  $1e-4$  to accommodate a fairly wide range of functions, which makes the second term  $1/\sqrt{n_l\lambda}$  dominate the risk if  $n_l$  is much smaller than  $1e4$ .

### 4.2.1 Connection with Domain Adaptation Learning Bounds

In Mansour et al. (2009), the authors provided several bounds on the pointwise difference of the loss for two different hypothesis (Theorem 11, 12 and 13). It is worth noting that in order to bound the pointwise loss, the authors make the following assumptions when the labeling function  $f_S$  (source) and  $f_T$  (target) are potentially different:

$$\delta^2 = L_{\hat{S}}(f_S(x), f_T(x)) \ll 1,$$



where  $L_{\hat{S}}(f_S(x), f_T(x)) = \mathbb{E}_{\hat{S}(x)} l(f_S(x), f_T(x))$ . This condition is easily violated under the model shift assumption, where the two labeling functions can differ by a large margin. However, with our transformation from  $Y^s$  to  $Y^{new}$ , we can translate the above assumption to the following equivalent condition:

$$\delta^2 = L_{\hat{S}}(Y^{new}, f_T(x)) = \frac{1}{m} \sum_{i=1}^m (y_i^{new} - \mathcal{U}[P_{Y^t|X^t}] \phi(x_i^s))^2$$

$$\leq (\epsilon\kappa + C'(\lambda_R^{1/2} + (n_l\lambda_R)^{-1/2}))^2,$$

using the results in Eq. 7. Hence we can bound  $\delta^2$  to be small under reasonable assumptions on  $n_l$  and  $\lambda_R$ .

#### 4.2.2 Comparing with No-transfer Learning

Without transfer, which means we predict on the unlabeled target set based merely on the labeled target set, the generalization error bound is simply:  $|\frac{1}{n_l} \sum_{i=1}^{n_l} l(h^{tL}, z_i^{tL}) - E_{z^t}[l(h^{tL}, z^t)]| \leq \frac{\sigma^2\kappa^2}{\lambda_t n_l} + (\frac{2\sigma^2\kappa^2}{\lambda_t} + 4M^2) \sqrt{\frac{\ln(1/\delta)}{2n_l}}$ , where  $h^{tL}$  is the KRR minimizer on  $\{X^{tL}, Y^{tL}\}$ . Then

$$E_{z^t}[l(h^{tL}, z^t)] - \frac{1}{n_l} \sum_{i=1}^{n_l} l(h^{tL}, z_i^{tL}) = O(\frac{1}{\lambda_t \sqrt{n_l}}). \quad (10)$$

We can see that with transfer learning, first we obtain a faster rate  $O(\lambda_t^{-1}(m + n_l)^{-1/2})$  in Eq. 9 with effective sample size  $n_l + m$  than  $O(\lambda_t^{-1}n_l^{-1/2})$  in Eq. 10 with effective sample size  $n_l$ . However, the transfer-rate Eq. 9 comes with a penalty term  $O(\lambda_R^{1/2} + (n_l\lambda_R)^{-1/2})$  which captures the estimation error between the transformed labels and the true target labels. Again, in transfer learning usually we assume  $m \rightarrow \infty$ , and  $n_l$  is relatively small, then the transfer-rate becomes  $O((n_l\lambda_R)^{-1/2})$ . Further if we assume that the smoothness parameter  $\lambda_R$  for the transformation is larger than the smoothness parameter  $\lambda_t$  for the target function ( $\lambda_R > \lambda_t$  will be sufficient if  $\lambda_R < 1$ , otherwise we need to set  $\lambda_R > \lambda_t^2$  if  $\lambda_R \geq 1$ ), then we obtain a faster convergence rate with transfer than no-transfer. We will further illustrate the results by empirical comparisons on synthetic and real data in the experimental section.

## 5 ANALYSIS ON THE OFFSET METHOD

In this section, we analyze the generalization error on the two-stage offset estimation (**Offset**) approach. Interestingly, our analysis shows that the generalization bounds for offset and CDM have the same dependency on  $n_l$ .

### 5.1 RISK ESTIMATES FOR OFFSET

(1) First, we learn a model from the source domain by minimizing the squared loss on the source data, i.e.,

$$h^s = \arg \min_{h \in \mathcal{H}} \frac{1}{m} \sum_{i=1}^m l(h, z_i^s) + \lambda_s \|h\|_k^2.$$

Using Theorem 1, we have with probability at least  $1 - \delta$ ,

$$R^s \leq R_{emp}^s + \frac{\sigma^2\kappa^2}{\lambda_s m} + (\frac{2\sigma^2\kappa^2}{\lambda_s} + 4M^2) \sqrt{\frac{\ln(1/\delta)}{2m}},$$

where  $R^s = E_{z^s}[l(h^s, z^s)]$ ,  $R_{emp}^s = \frac{1}{m} \sum_{i=1}^m l(h^s, z_i^s)$ . Hence

$$R^s - R_{emp}^s = O(\frac{1}{\lambda_s \sqrt{m}}), \quad (11)$$

(2) Second, we learn the offset by KRR on  $\{X^{tL}, \hat{y}^o\}$ , where  $\hat{y}^o = Y^{tL} - f^s(X^{tL})$ , i.e.,  $\hat{y}^o$  is the estimated offset on labeled target points  $X^{tL}$ , and  $f^s(X^{tL})$  is the prediction on  $X^{tL}$  using source data.

Denote  $\hat{h}^o$  as the minimizer on  $\hat{z}^o = \{X^{tL}, \hat{y}^o\}$ , i.e.,

$$\hat{h}^o = \arg \min_{h \in \mathcal{H}} \frac{1}{n_l} \sum_{i=1}^{n_l} l(h, \hat{z}_i^o) + \lambda_o \|h\|_k^2$$

$$= \arg \min_{h \in \mathcal{H}} R(h) + N(h). \quad (12)$$

Denote  $h^o$  as the minimizer on  $z^o = \{X^{tL}, y^o\}$ , where  $y^o$  is the unknown true offset:

$$h^o = \arg \min_{h \in \mathcal{H}} \frac{1}{n_l} \sum_{i=1}^{n_l} l(h, z_i^o) + \lambda_o \|h\|_k^2$$

$$= \arg \min_{h \in \mathcal{H}} R'(h) + N(h), \quad (13)$$

Using Theorem 1, we have with probability at least  $1 - \delta$ ,

$$R^o \leq R_{emp}^o + \frac{\sigma^2\kappa^2}{\lambda_o n_l} + (\frac{2\sigma^2\kappa^2}{\lambda_o} + 4M^2) \sqrt{\frac{\ln(1/\delta)}{2n_l}} \quad (14)$$

where  $R^o = E_{z^o}[l(h^o, z^o)]$ ,  $R_{emp}^o = \frac{1}{n_l} \sum_{i=1}^{n_l} l(h^o, z_i^o)$ . In our estimation we use  $\hat{y}^o$  instead of  $y^o$ , hence we need to account for this estimation error.

**Lemma 1.** *The generalization error  $R^o$  is bounded by:*

$$R^o = \bar{R}_{emp}^o + O(\frac{1}{\lambda_o \sqrt{n_l}}), \quad (15)$$

as  $m \rightarrow \infty$ . Here  $\bar{R}_{emp}^o = \frac{1}{n_l} \sum_{i=1}^{n_l} l(\hat{h}^o, \hat{z}_i^o)$  is the empirical error of our estimator  $\hat{h}^o$  on  $\{X^{tL}, \hat{y}^o\}$ .

*Proof.* Define the Bregman Divergence associated to  $F$  of  $f$  to  $g$  by  $B_F(f||g) = F(f) - F(g) - \langle f - g, \nabla F(g) \rangle$ . Let  $F(h) = R(h) + N(h)$ ,  $F'(h) = R'(h) + N(h)$ . Since  $h^o, \hat{h}^o$  are the minimizers, we have  $B_{F'}(\hat{h}^o||h^o) + B_F(h^o||\hat{h}^o) = F'(\hat{h}^o) - F'(h^o) + F(h^o) - F(\hat{h}^o) = R'(\hat{h}^o) - R'(h^o) + R(h^o) - R(\hat{h}^o)$ . In addition, using the nonnegativity of  $B$  and  $B_F = B_R + B_N$ ,  $B_{F'} = B_{R'} + B_N$ , we have  $B_N(\hat{h}^o||h^o) + B_N(h^o||\hat{h}^o) \leq B_F(h^o||\hat{h}^o) + B_{F'}(\hat{h}^o||h^o)$ . Combining the two we have  $B_N(\hat{h}^o||h^o) + B_N(h^o||\hat{h}^o) \leq R'(\hat{h}^o) - R'(h^o) +$

$$R(h^o) - R(\hat{h}^o) = \frac{1}{n_l} \sum_{i=1}^{n_l} l(\hat{h}^o, z_i^o) - \frac{1}{n_l} \sum_{i=1}^{n_l} l(h^o, z_i^o) + \frac{1}{n_l} \sum_{i=1}^{n_l} l(h^o, \hat{z}_i^o) - \frac{1}{n_l} \sum_{i=1}^{n_l} l(\hat{h}^o, \hat{z}_i^o) \leq \frac{2}{n_l} \sum_{i=1}^{n_l} \sigma |y_i^o - \hat{y}_i^o|, \text{ using } |l(\hat{h}^o, z_i^o) - l(\hat{h}^o, \hat{z}_i^o)| \leq |2\hat{h}^o(x_i) - y_i^o - \hat{y}_i^o| \cdot |y_i^o - \hat{y}_i^o| \leq \sigma |y_i^o - \hat{y}_i^o|, \sigma = 4M.$$

Since for RKHS norm  $B_N(f||g) = \|f - g\|_k^2$ , we have  $B_N(\hat{h}^o||h^o) + B_N(h^o||\hat{h}^o) = 2\|h^o - \hat{h}^o\|_k^2$ . Combined with the above inequality, we have  $2\|h^o - \hat{h}^o\|_k^2 \leq \frac{2}{n_l} \sum_{i=1}^{n_l} \sigma |y_i^o - \hat{y}_i^o|$ . Then we have  $|l(h^o, z_i^o) - l(\hat{h}^o, z_i^o)| \leq \sigma |h^o(x_i) - \hat{h}^o(x_i)| \leq \sigma \|h^o - \hat{h}^o\|_k \kappa \leq \sigma \kappa \sqrt{\frac{1}{n_l} \sum_{i=1}^{n_l} \sigma |y_i^o - \hat{y}_i^o|}$ . Hence  $|\frac{1}{n_l} \sum_{i=1}^{n_l} l(h^o, z_i^o) - \frac{1}{n_l} \sum_{i=1}^{n_l} l(\hat{h}^o, \hat{z}_i^o)| \leq \frac{1}{n_l} \sum_{i=1}^{n_l} [|l(h^o, z_i^o) - l(\hat{h}^o, z_i^o)| + |l(\hat{h}^o, z_i^o) - l(\hat{h}^o, \hat{z}_i^o)|] \leq \sigma \kappa \sqrt{\frac{1}{n_l} \sum_{i=1}^{n_l} \sigma |y_i^o - \hat{y}_i^o|} + \frac{1}{n_l} \sum_{i=1}^{n_l} \sigma |y_i^o - \hat{y}_i^o|$ . Now we can conclude that

$$R_{emp}^o = \frac{1}{n_l} \sum_{i=1}^{n_l} l(h^o, z_i^o) \leq \bar{R}_{emp}^o + P, \quad (16)$$

where  $\bar{R}_{emp}^o = \frac{1}{n_l} \sum_{i=1}^{n_l} l(\hat{h}^o, \hat{z}_i^o)$ , and  $P = \sigma \kappa \sqrt{\frac{1}{n_l} \sum_{i=1}^{n_l} \sigma |y_i^o - \hat{y}_i^o|} + \frac{1}{n_l} \sum_{i=1}^{n_l} \sigma |y_i^o - \hat{y}_i^o|$ . To bound  $P$ , first we have  $\frac{1}{n_l} \sum_{i=1}^{n_l} |y_i^o - \hat{y}_i^o| = \frac{1}{n_l} \sum_{i=1}^{n_l} |(y_i^{tL} - y_i^s) - (y_i^{tL} - \hat{y}_i^s)| = \frac{1}{n_l} \sum_{i=1}^{n_l} |y_i^s - \hat{y}_i^s| \leq \sqrt{\frac{1}{n_l} \sum_{i=1}^{n_l} (y_i^s - \hat{y}_i^s)^2}$ . Using Eq. 11,  $\frac{1}{n_l} \sum_{i=1}^{n_l} (y_i^s - \hat{y}_i^s)^2$  is bounded by  $R_{emp}^s + O(\frac{1}{\lambda_s \sqrt{m}})$ . We can see that the penalty term  $P$  diminishes as  $m \rightarrow \infty$ . Plugging Eq. 16 into Eq. 14 concludes the proof.  $\square$

(3) Now we analyze the generalization error in the target domain. Using the assumption that the target labels  $y^t$  can also be decomposed by  $y^o + y^s$ , we have:

$$\begin{aligned} \mathbb{E}_{z^t} [l(h, z^t)] &= \mathbb{E}_{z^t} [(h(x^t) - y^t)^2] \\ &= \mathbb{E}[(h^o(x^t) + h^s(x^t) - y^o - y^s)^2] \\ &\leq 2\mathbb{E}(h^o(x^t) - y^o)^2 + 2\mathbb{E}(h^s(x^t) - y^s)^2. \end{aligned} \quad (17)$$

Plugging in Eq. 11 and Eq. 15, we have

$$R^t = \mathbb{E}_{z^t} [l(h, z^t)] = 2R_{emp}^s + 2\bar{R}_{emp}^o + O\left(\frac{1}{\lambda_o \sqrt{n_l}} + \frac{1}{\lambda_s \sqrt{m}}\right)$$

In transfer learning usually we assume that the number of source data is sufficient, i.e.,  $m \rightarrow \infty$ , hence

$$R^t - 2(R_{emp}^s + \bar{R}_{emp}^o) = O\left(\frac{1}{\lambda_o \sqrt{n_l}}\right). \quad (18)$$

### 5.1.1 Comparing with No-transfer Learning

As with the no-transfer-rate in Sec. 4.2.2, we have

$$R^t - R_{emp}^{tL} = O\left(\frac{1}{\lambda_t \sqrt{n_l}}\right), \quad (19)$$

where  $\lambda_t$  is the regularization parameter when estimating the target function. Comparing this rate with Eq. 18, and using our assumption that we have a smoother offset than the target function, i.e.,  $\lambda_o > \lambda_t$ , we can see that we obtain a faster convergence rate with transfer than no-transfer.

## 6 MULTI-SOURCE TRANSFER LEARNING

In this section, we show that we can easily adapt the transfer learning algorithm from a single source to transfer learning with multiple-sources, by utilizing the generalization bounds we derived in earlier sections. Transfer learning with multiple sources is similar to multi-task learning, where we learn the target and multiple sources jointly.

A closer look at Eq. 9 for CDM, and Eq. 18 for Offset reveals that, when  $n_l$  is small and  $m \rightarrow \infty$ , we have a convergence rate of  $O(\frac{1}{\lambda_* \sqrt{n_l}})$  for both algorithms, where  $\lambda_*$  is some parameter that controls the smoothness of the source-to-target transfer (for Eq. 9 we can set  $\lambda_R = \lambda_*^2$ ). This observation motivates our reweighting scheme on the source hypotheses to achieve transfer learning under multiple sources, described as the following.

Assume we have  $S$  sources and a target. First, we apply the transfer learning algorithm from a single source to obtain a model  $M_s$  from each source  $s$  to target  $t$ , where the parameter  $\lambda_*^s$  is determined by cross-validation,  $s = 1, \dots, S$ . Second, we compute the weight for each source  $s$  by:

$$\begin{aligned} w_s &= p(D|M_s)p(M_s), \text{ where} \\ p(D|M_s) &= \exp\left\{-\sum_{i=1}^{m_s} (y_i^{tL} - \hat{f}^s(x_i^{tL}))^2\right\}, \\ p(M_s) &\propto \exp\left\{-\alpha \frac{1}{\lambda_*^s}\right\}, \end{aligned}$$

where  $\hat{f}^s(x_i^{tL})$  is the prediction given by  $M_s$ .

The idea is similar to Bayesian Model Averaging (Hoeting et al. (1999)), where the first term  $p(D|M_s)$  serves as the data likelihood of the predictive model  $M_s$  from source  $s$ , and the second term  $p(M_s)$  is the prior probability on model  $M_s$ . In our case,  $p(M_s)$  is chosen to indicate how similar each source to the target is, where the similarity is measured by how smooth the change is from source  $s$  to target  $t$ . It is easy to see that, the weights coincide with our analysis of the generalization bounds for transfer learning, and the choice of  $\alpha$  should be in the order of  $O(1/\sqrt{n_l})$ . Intuitively, when the number of labeled target points  $n_l$  is small,  $p(M_s)$  has a larger effect on  $w_s$ , which means we prefer the source that has a smoother change (larger  $\lambda_*^s$ ) for the transfer. On the other hand, when  $n_l$  is large, then  $p(D|M_s)$  takes over, i.e., we prefer the source that results in a larger data likelihood (smaller prediction errors). Fi-

nally, we combine the predictions by:

$$\hat{f}(x_i^{tU}) = \sum_{s=1}^S \frac{w_s}{\sum_{s=1}^S w_s} \hat{f}^s(x_i^{tU})$$

This weighted combination of source hypotheses gives us the following generalization bound in the target domain:

$$\begin{aligned} \mathbb{E}_{z^t} [l(h, z^t)] &= \mathbb{E}_{z^t} \left[ \left( \sum_s \frac{w_s}{\sum_{s=1}^S w_s} h_s(x^t) - y^t \right)^2 \right] \\ &= \mathbb{E}_{z^t} \left[ \left( \sum_s \frac{w_s}{\sum_{s=1}^S w_s} (h_s(x^t) - y^t) \right)^2 \right] \\ &\leq \sum_s \frac{w_s}{\sum_{s=1}^S w_s} \mathbb{E}_{z^t} [(h_s(x^t) - y^t)^2] \\ &= \sum_s \frac{w_s}{\sum_{s=1}^S w_s} [\tilde{R}_{emp}^s + O(\frac{1}{\lambda_s \sqrt{n_t}})], \end{aligned}$$

where the third inequality uses Jensen’s inequality, and the last equality uses the bounds we derived. Here  $\tilde{R}_{emp}^s$  refers to the empirical error for source  $s$  when transferring from  $s$  to  $t$  (Thm. 2 for CDM and Eq. 18 for Offset).

## 7 EXPERIMENTS

### 7.1 SYNTHETIC EXPERIMENTS

In this section, we empirically compare the generalization error of transfer learning algorithms to that of no-transfer learning (learning by labeled target data only), on synthetic datasets simulating different conditions.

We generate the synthetic dataset in this way:  $X^s, X^t$  are drawn uniformly at random from  $[0, 4]$ ,  $Y^s = \sin(2X^s) + \sin(3X^s)$  with additive Gaussian noise.  $Y^t$  is the same function with a smoother location-scale transformation/offset. In each of the following comparisons, we plot the mean squared error (MSE) on the unlabeled target points (as an estimation of the generalization error) with respect to different number of labeled target points. The labeled target points are chosen uniformly at random, and we average the error over 10 experiments. The parameters are chosen using cross validation.

In Fig. 3, we compare transfer learning using CDM with no-transfer learning. The results show that with the additional smoothness assumption, we are able to achieve a much lower generalization error for transfer learning than no-transfer learning. In Fig. 4 and 5, we compare transfer learning using the Offset approach with no-transfer learning. The two figures show different generalization error curves when the smoothness of the offset is different. We can see that with a smoother offset (Fig. 4) we are able to achieve a much lower generalization error than no-transfer learning. With a less smooth offset (Fig. 5) we can still achieve a lower generalization error than no-transfer learning, but the rate is slower compared to Fig. 4. Further we

analyze the case when the smoothness assumption does not hold, by setting the source function to be  $\sin(X^s) + \epsilon$  such that the target changes faster than the source. In this case, transfer learning with CDM/Offset yield almost the same generalization error as no-transfer learning (Fig. 6), i.e., the source data does not help in learning the target.

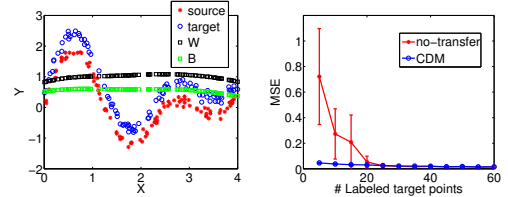


Figure 3: No-transfer learning vs. transfer learning (CDM)

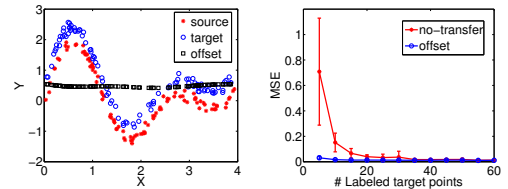


Figure 4: No-transfer learning vs. transfer learning using the Offset approach (smoother offset,  $\lambda_R = 0.1$ )

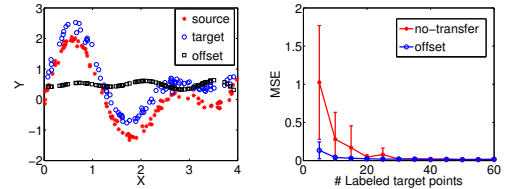


Figure 5: No-transfer learning vs. transfer learning using the Offset approach (less smooth offset,  $\lambda_R = 0.001$ )

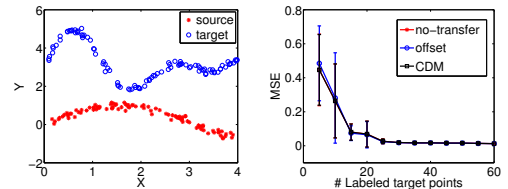


Figure 6: No-transfer learning vs. transfer learning, when the smoothness assumption does not hold

### 7.2 EXPERIMENTS ON THE REAL DATA

#### 7.2.1 Comparing Transfer Learning to No-transfer Learning, Using Different Sources

The real-world dataset is an Air Quality Index (AQI) dataset (Mei et al. (2014)) during a 31-day period from Chinese cities. For each city, the input feature  $x_i$  is a bag-of-words vector extracted from Weibo posts of each day, with 100,395 dimensions as the dictionary size. The output label  $y_i$  is a real number which is the AQI of that day.

Fig. 7 shows a comparison of MSE on the unlabeled target points, with respect to different number of labeled target

points, when transferring from a nearby city (Ningbo) and a faraway city (Xi'an), to a target city (Hangzhou). The data is shown in the left figure of Fig. 7, where the x-axis is each day. The results are averaged over 20 experiments with uniformly randomly chosen labeled target points. First we observe that we obtain a lower generalization error by transferring from other cities than learning by the target city data alone (no-transfer). In addition, the generalization error are much lower if we transfer from nearby cities where the difference between source and target is smoother.

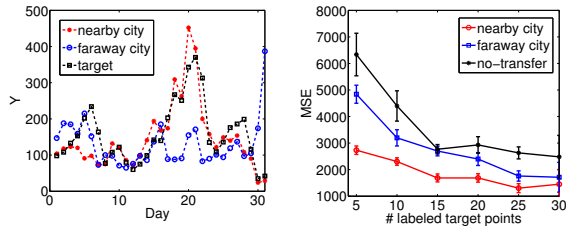


Figure 7: Comparison of MSE on unlabeled target points

### 7.2.2 Transfer Learning with Multiple Sources

The results in Sec. 7.2.1 indicate that, when transferring from multiple sources to a target, it is important to choose which source to transfer, in order to obtain a larger gain. In this section, we show the results on the same air quality index data by reweighting different sources (Sec. 6).

Fig. 8 shows a comparison of MSE on the unlabeled target data (data shown in the left figure) with respect to different number of labeled target points ( $n_l \in \{2, 5, 10, 15, 20\}$ ), where the prediction is based on each source independently (labeled as **source**  $i$ ,  $i \in \{1, 2, 3\}$ ), and based on multiple sources (labeled as **posterior**). Since CDM and Offset give similar bounds, we use two-stage offset estimation as the prediction algorithm from each source  $s$  to target  $t$ . The weighting on the sources is as described in Sec. 6. As can be seen from the results, using posterior reweighting on different sources, we obtain results that are very close to the results using the best source.

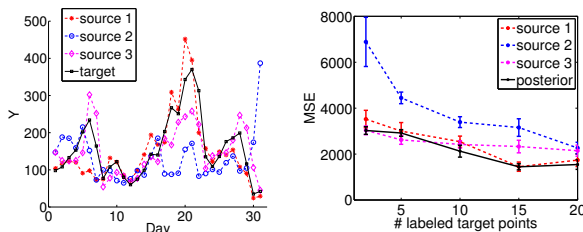


Figure 8: Comparison of MSE on unlabeled target points, with multiple sources

Further in Figure. 9, we show a comparison of MSE on the unlabeled target data between the proposed approach and two baselines, with respect to different number of la-

beled target points. The results are averaged over 20 experiments. The first baseline **wDA** is a weighted multi-source domain adaptation approach proposed in Mansour et al. (2008), where the distribution  $D_i(x)$  for source  $i$  on a target point  $x$  is estimated using kernel density estimation with a Gaussian kernel. Note that the original algorithm proposed in Mansour et al. (2008) does not assume the existence of a few labeled target points, thus the hypothesis  $h_i(x)$  from each source  $i$  is computed by using the source data only. To ensure a fair comparison, we augment  $h_i(x)$  by using the prediction of the Offset approach given  $n_l$  labeled target points. The second baseline **optDA** is a multi-source domain adaptation algorithm under an optimization framework, as proposed in Chattopadhyay et al. (2011), where the parameters  $\gamma_A, \gamma_I$  are set as described in the paper, and  $\theta$  is chosen using cross-validation on the set  $\{0.1, 0.2, \dots, 0.9\}$  (the final choice of  $\theta$  is 0.1). Note that our proposed algorithm gives the best performance. In addition, our algorithm does not require density estimation as in **wDA**, which can be difficult in real-world applications with high-dimensional features. Further note **posterior** considers the change in  $P(Y|X)$  while **wDA** focuses on the change of  $P(X)$ . A potential improvement can be achieved by combining these two in the reweighting scheme, which should be an interesting future direction.

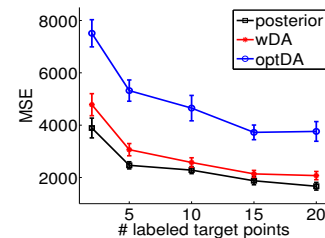


Figure 9: Multi-source transfer learning: comparison of MSE on the proposed approach (**posterior**) and baselines

## 8 CONCLUSION

In this paper, we provide theoretical analysis for algorithms proposed for transfer learning under the model shift assumption. Unlike previous work on covariate shift, the model shift poses a harder problem for transfer learning, and our analysis shows that we are still able to achieve a similar rate as in covariate shift/domain adaptation, modified by the smoothness of the transformation parameters. We also show conditions when transfer learning works better than no-transfer learning. Finally we extend the algorithms to transfer learning with multiple sources.

### Acknowledgements

This work is supported in part by the US Department of Agriculture under grant number 20126702119958.

## References

- Yishay Mansour, Mehryar Mohri, and Afshin Ros-tamizadeh. Domain Adaptation with multiple sources. *NIPS* 2008.
- Yishay Mansour, Mehryar Mohri, and Afshin Ros-tamizadeh. Domain Adaptation: Learning Bounds and Algorithms. *COLT* 2009.
- Olivier Bousquet and Andre Elisseeff. Stability and Generalization. *JMLR* 2002.
- Jiayuan Huang, Alex Smola, Arthur Gretton, Karsten Borgwardt, and Bernhard Schölkopf. Correcting sample selection bias by unlabeled data. *NIPS* 2007, 2007.
- Arthur Gretton, Karsten M. Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alex Smola. A kernel method for the two-sample-problem. *NIPS* 2007, 2007.
- Xuezhi Wang, Tzu-Kuo Huang, and Jeff Schneider. Active transfer learning under model shift. *ICML*, 2014.
- S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira. Analysis of representations for domain adaptation. *NIPS* 2006.
- J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. Wortman. Learning bounds for domain adaptation. *NIPS* 2007.
- Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *TKDE* 2009, 2009.
- Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90 (2): 227-244, 2000.
- Lilyana Mihalkova, Tuyen Huynh, and Raymond J. Mooney. Mapping and revising markov logic networks for transfer learning. *Proceedings of the 22nd AAAI Conference on Artificial Intelligence (AAAI-2007)*, 2007.
- Chuong B. Do and Andrew Y. Ng. Transfer learning for text classification. *Neural Information Processing Systems Foundation*, 2005.
- Rajat Raina, Andrew Y. Ng, and Daphne Koller. Constructing informative priors using transfer learning. *Proceedings of the Twenty-third International Conference on Machine Learning*, 2006.
- Alexandru Niculescu-Mizil and Rich Caruana. Inductive transfer for bayesian network structure learning. *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2007.
- J. Jiang and C. Zhai. Instance weighting for domain adaptation in nlp. *Proc. 45th Ann. Meeting of the Assoc. Computational Linguistics*, pp. 264-271, 2007.
- X. Liao, Y. Xue, and L. Carin. Logistic regression with an auxiliary data source. *Proc. 21st Intl Conf. Machine Learning*, 2005.
- Kun Zhang, Bernhard Schölkopf, Krikamol Muandet, and Zhikun Wang. Domain adaptation under target and conditional shift. *ICML 2013*, 2013.
- Le Song, Jonathan Huang, Alex Smola, and Kenji Fukumizu. Hilbert space embeddings of conditional distributions with applications to dynamical systems. *ICML 2009*, 2009.
- Rita Chattopadhyay, Jieping Ye, Sethuraman Panchanathan, Wei Fan, and Ian Davidson. Multi-Source Domain Adaptation and Its Application to Early Detection of Fatigue. *KDD'11*, 2011.
- Yi Yao and Gianfranco Doretto. Boosting for transfer learning with multiple sources. *CVPR 2010*, 2010.
- Koby Crammer, Michael Kearns, and Jennifer Wortman. Learning from Multiple Sources. *JMLR* 2008, 2008.
- Shike Mei, Han Li, Jing Fan, Xiaojin Zhu, and Charles R. Dyer. Inferring Air Pollution by Sniffing Social Media. *The International Conference on Advances in Social Network Analysis and Mining (ASONAM)*, 2014.
- Jennifer A. Hoeting, David Madigan, Adrian E. Raftery, and Chris T. Volinsky. Bayesian Model Averaging: A Tutorial. *Statistical Science*, Vol. 14, No. 4, 382-417, 1999.

---

# Clustered Sparse Bayesian Learning

---

Yu Wang \*

David Wipf †

Jeong-Min Yun ‡

Wei Chen\*

Ian Wassell \*

\* University of Cambridge, Cambridge, UK

† Microsoft Research, Beijing, China

‡ Pohang University of Science and Technology, Pohang, Republic of Korea

yw323@cam.ac.uk davidwip@microsoft.com azida@postech.ac.kr wc253@cam.ac.uk ijw24@cam.ac.uk

## Abstract

Many machine learning and signal processing tasks involve computing sparse representations using an overcomplete set of features or basis vectors, with compressive sensing-based applications a notable example. While traditionally such problems have been solved individually for different tasks, this strategy ignores strong correlations that may be present in real world data. Consequently there has been a push to exploit these statistical dependencies by jointly solving a series of sparse linear inverse problems. In the majority of the resulting algorithms however, we must a priori decide which tasks can most judiciously be grouped together. In contrast, this paper proposes an integrated Bayesian framework for both clustering tasks together and subsequently learning optimally sparse representations within each cluster. While probabilistic models have been applied previously to solve these types of problems, they typically involve a complex hierarchical Bayesian generative model merged with some type of approximate inference, the combination of which renders rigorous analysis of the underlying behavior virtually impossible. On the other hand, our model subscribes to concrete motivating principles that we carefully evaluate both theoretically and empirically. Importantly, our analyses take into account all approximations that are involved in arriving at the actual cost function to be optimized. Results on synthetic data as well as image recovery from compressive measurements show improved performance over existing methods.

---

Y. Wang is sponsored by the University of Cambridge Overseas Trust. Y. Wang and J. Yun are partially supported by sponsorship from Microsoft Research Asia. W. Chen is supported by EPSRC Research Grant EP/K033700/1 and the NSFC Research Grant 61401018.

## 1 INTRODUCTION

Solving sparse linear inverse problems is a fundamental building block in numerous machine learning, computer vision, and signal processing applications related to compressive sensing and beyond (Elhamifar & Vidal 2013; Soltanolkotabi & Candes, 2012; Zhang & Rao, 2011; Hu et al., 2013). In its most basic form, sparse estimation algorithms are built upon the observation model

$$\mathbf{y} = \Phi \mathbf{x} + \epsilon, \quad (1)$$

where  $\Phi \in \mathbb{R}^{N \times M}$  is a dictionary of basis vectors that we assume to have unit  $\ell_2$  norm,  $\mathbf{x} \in \mathbb{R}^M$  is a vector of unknown coefficients we would like to estimate,  $\mathbf{y} \in \mathbb{R}^N$  is an observed measurement vector, and  $\epsilon$  is a noise vector distributed as  $\mathcal{N}(0, \nu I)$ . The objective is to estimate the unknown generative  $\mathbf{x}$  under the assumption that it is maximally sparse, meaning that  $\|\mathbf{x}\|_0$  is minimal. Here  $\|\cdot\|_0$  represents the canonical  $\ell_0$  norm sparsity metric, or a count of the number of nonzero elements in a vector. This sparse linear inverse problem is compounded considerably by the additional assumption that  $M > N$ , meaning the dictionary  $\Phi$  is *overcomplete*.

Now suppose that we have access to multiple measurement vectors from  $L$  different tasks of interest that are assembled as  $Y = [\mathbf{y}_1, \dots, \mathbf{y}_L] \in \mathbb{R}^{N \times L}$  and linked to a corresponding matrix of unknown coefficients  $X = [\mathbf{x}_1, \dots, \mathbf{x}_L] \in \mathbb{R}^{M \times L}$  via

$$\mathbf{y}_j = \Phi_j \mathbf{x}_j + \epsilon_j, \quad \forall j = 1, \dots, L. \quad (2)$$

If these measurement vectors and associated coefficients maintain some degree of dependency, for example the locations of zero-valued elements (or support sets) are statistically related, then it will generally be advantageous to jointly estimate  $X$  from  $Y$  as opposed to solving for each  $\mathbf{x}_j$  individually (Obozinski et al., 2011). Perhaps the simplest example of this is frequently referred to as simultaneous sparse approximation (Tropp, 2006) or multi-task compressive sensing (Ji et al., 2009). In brief, these paradigms follow from the assumption that  $X$  is maximally *row sparse*, implying that each column of  $X$  is maximally sparse with a common support pattern. This model has been

applied to compressive sensing of images and video (Ji et al., 2009), tracking (Hong et al., 2013), and medical image analysis (Wan et al., 2012). Moreover, it can be naturally extended to handle a richer set of dependencies by applying a known graph structure that groups subsets of columns together for joint estimation (Cevher et al., 2008; Shi et al., 2014), as opposed to strict enforcement of row sparsity across all measurements. Other pre-defined structural assumptions can be found in (Archambeau et al., 2011; Jalali et al., 2013; Rao et al., 2013; Yang & Ravikumar, 2013).

The limitation of all of these strategies is that they are predicated on prior knowledge of how tasks should be grouped together to optimally facilitate subsequent sparse estimation. In contrast, here we intend to develop a principled multi-task learning algorithm that simultaneously clusters tasks blindly while optimally estimating sparse coefficient vectors informed by these clusters. We should state at the outset that multi-task compressive sensing has been merged with cluster learning before (Qi et al., 2008). However, this algorithm relies on a complex hierarchical model anchored with an approximate Dirichlet process prior distribution on  $X$  (Blei & Jordan, 2006). Subsequent model inference then requires an additional variational mean-field approximation. Overall the fundamental underlying mechanics of the model have not been carefully analyzed nor understood, nor is there any guarantee that sparsity will necessarily result. Other even more complex hierarchical models have been applied to somewhat-related multi-task learning problems, e.g., (Hernandez-Lobato & Hernandez-Lobato, 2013); however, these models require complex inference procedures that must ultimately be justified by the validity of the assumed prior distributions rather than provable properties of the resulting estimators.

The remainder of the paper is organized as follows. In Section 2 we first motivate our design principles. A specific Bayesian model and corresponding objective function are then developed in Section 3. Next we derive updates rules for optimization purposes, leading to our *clustered sparse Bayesian learning algorithm* (C-SBL) in Section 4. We theoretically and empirically analyze this framework in Sections 5 and 6 respectively, revealing that it is consistent with our original motivational principles. Moreover, estimation results on synthetic data and real images demonstrate improved estimation quality relative to existing algorithms when using compressive measurements. Overall, we summarize our contributions as follows:

- Analysis of specific, previously-unexamined theoretical principles that play a critical role in multi-task sparse estimation problems.
- Development of a robust sparse Bayesian algorithm that adheres to these principles to an extent not seen in any existing algorithm we are aware of.
- Although we employ a Bayesian entry point for our

algorithmic strategy, final model justification is provided entirely based on rigorous properties of the underlying cost function that emerges, *including all approximations involved*, rather than any putative belief in the actual validity of assumed prior distributions.

## 2 MOTIVATING PRINCIPLES

When deriving an algorithm for joint clustering and multi-task sparse estimation, it is helpful to first define a few basic attributes that ideally any procedure might possess. Here we consider properties related to limiting behavior as the noise variance  $\nu$  varies from zero towards large values for each task.

First assume  $\nu \rightarrow 0$  and the following generative model. Let  $X^*$  denote the true coefficient matrix we would like to estimate using measurements  $\mathbf{y}_j = \Phi_j \mathbf{x}_j^*$ . We assume that the columns of  $X^*$  are partitioned into clusters with common sparsity profile or support within each cluster. Additionally let  $\Omega_k$  denote the column indices of  $X^*$  associated with cluster  $k = 1, \dots, K \leq L$ . For any matrix  $Z$  define  $Z_{\Omega_k}$  as the sub-matrix of columns associated with the index set  $\Omega_k$ . Then the relevant sparse linear inverse problem, assuming known clusters, becomes

$$\min_{\{X_{\Omega_k}\}} \sum_k |\Omega_k| \|X_{\Omega_k}\|_{row-\ell_0} \quad \text{s.t. } \mathbf{y}_j = \Phi_j \mathbf{x}_j, \forall j, \quad (3)$$

where  $\|\cdot\|_{row-\ell_0}$  counts the number of nonzero rows of a matrix, which is then weighted by the cardinality of the set  $\Omega_k$  in the objective function.<sup>1</sup>

Now assume the perturbation model

$$\bar{X}_k^* = A_k^x + \alpha^x R_k^x, \forall k, \quad (4)$$

where  $\bar{X}_{\Omega_k}^*$  denotes the nonzero rows of  $X_{\Omega_k}^*$  associated with cluster  $k$ ,  $A_k^x$  is an arbitrary matrix of appropriate dimensions,  $\alpha^x > 0$  is an arbitrarily small scalar, and  $R_k^x$  is a random matrix with iid, continuously-distributed elements. Likewise, assume that

$$\Phi_j = A_j^\phi + \alpha^\phi R_j^\phi, \forall j, \quad (5)$$

with analogous definitions to those in (4), albeit with obviously different dimensions and values. Then we have the following:

**Lemma 1.** Suppose we are given any  $Y$  generated with  $\mathbf{y}_j = \Phi_j \mathbf{x}_j^*$  and  $\|\mathbf{x}_j^*\|_0 < N \forall j$ , where  $X^*$  satisfies (4)  $\forall k$  and  $\Phi_j$  satisfies (5)  $\forall j$ . Then  $X^*$  is the unique global minimum of both (3) and

$$\min_X \sum_j \|\mathbf{x}_j\|_0 \quad \text{s.t. } \mathbf{y}_j = \Phi_j \mathbf{x}_j, \forall j. \quad (6)$$

<sup>1</sup>Actually, this weighting factor is irrelevant to what follows in the noiseless case, but does play a role later when we consider noisy conditions.

The proof is relatively straightforward and comes from modifying Theorem 1 from (Baron et al., 2009). While perhaps notationally cumbersome to present, the message of this result is simple and widely applicable. In words, Lemma 1 implies that, under general conditions that apply to virtually any multi-task system of interest (since the sets  $\{A_k^x\}$   $\{A_j^\phi\}$  are arbitrary and the perturbations applied to them can be infinitesimally small), the unique maximally row-sparse solution to the clustering problem is equivalent to the global solution obtained by simply evaluating each task individually. The cluster structure itself does not provide any direct advantage, and we could just as well solve (6), and without noise we can theoretically resolve any number of clusters between 1 and  $L$ .

With this in mind then, in the limit  $\nu \rightarrow 0$  we would prefer to have a clustering algorithm whose global optimum is equivalent to (6). However, we need not solve this problem directly, which in general is NP-hard. Rather we favor an algorithm that can, to the extent possible, leverage cluster information to steer the algorithm towards the global solution of (6) while avoiding bad local optima.

Now consider larger values of noise, i.e.,  $\nu > 0$ , where we would like to solve something akin to

$$\min_X \sum_j \|\mathbf{y}_j - \Phi_j \mathbf{x}_j\|_2^2 + \nu \sum_k |\Omega_k| \|X_{\Omega_k}\|_{\text{row-}\ell_0}. \quad (7)$$

In general, when the noise level is high we cannot hope to resolve a large number of clusters, and eventually we must merge to a single cluster as  $\nu$  becomes sufficiently large. In this regime the issue is not so much one of avoiding bad local minima as it is enhancing the effective signal-to-noise ratio as much as possible. Note that the smaller  $K$  is, the more tasks per cluster, which has a substantial benefit in terms of signal-to-noise ratio. This can be easily visualized via the special case where  $\Phi_j^\top \Phi_j = I \forall j$ . Given this simplification, (7) has a closed-form solution given by

$$x_{i,j}^* = z_{i,j} \mathcal{I} \left[ \sum_{j \in \Omega_{c(j)}} z_{i,j}^2 > \nu |\Omega_{c(j)}| \right], \quad (8)$$

where  $\mathbf{z}_j \triangleq \Phi_j^\top \mathbf{y}_j$ ,  $c(j)$  denotes the cluster index of task  $j$ , and  $\mathcal{I}$  is an indicator function. Thus the optimal solution represents a hard-thresholding operation, where the threshold is dictated by an average across tasks within each cluster. If we have only a single cluster, meaning  $c(j) = 1 \forall j$  and  $\Omega_1 = \{1, \dots, L\}$ , then this threshold value is maximally robust to noise given that all tasks are averaged together to increase the SNR of the threshold.

To conclude then, there are (at least) three important considerations:

1. At high SNR, local minima avoidance while finding maximally sparse solutions is paramount. We would also favor that, for a given clustering, maximally row-sparse solutions can be obtained by evading any sub-optimal local extrema where possible.

2. At low SNR when it is impossible to resolve many clusters anyway, the issue is more about merging clusters to hopefully improve the implicit SNR.
3. In intermediate regimes we would like to accomplish a bit of both.

In Section 5 we provide theoretical evidence that our proposed algorithm is favorable with respect to points 1 and 2, while Section 6 presents empirical evidence in practical support of point 3.

### 3 MODEL DESCRIPTION

While perhaps not immediately obvious at first, this section will develop a Bayesian model that ultimately reflects the previously stated principles. Consistent with the observation model in (1), we adopt the Gaussian likelihood function

$$p(Y|X) \propto \prod_j \exp \left[ -\frac{1}{2\nu} \|\mathbf{y}_j - \Phi_j \mathbf{x}_j\|_2^2 \right]. \quad (9)$$

For present purposes we will assume that the noise variance  $\nu$  is known (ultimately though this value can be learned from the data). For the prior distribution on each  $\mathbf{x}_j$  we build upon the basic sparse Bayesian learning framework from (Tipping, 2001) which in the present context would involve a zero-mean Gaussian with an independent diagonal covariance; however, this would not allow for task clustering. Instead we assume the prior distribution

$$p(X|\Lambda, W) \propto \prod_j \exp \left[ -\frac{1}{2} \mathbf{x}_j^\top \Gamma_j^{-1} \mathbf{x}_j \right], \quad (10)$$

where  $\Lambda \in \mathbb{R}^{M \times K}$  and  $W \in \mathbb{R}^{L \times K}$  are hyperparameter matrices;  $\Lambda$  is constrained to have all non-negative elements,  $W \in \mathcal{S}$  is defined such that each row denoted as  $\mathbf{w}^j$  is an element of the probability simplex, i.e.,

$$\mathcal{S} \triangleq \{\mathbf{w}^j : \sum_k w_{j,k} = 1, w_{j,k} \in [0, 1]\}. \quad (11)$$

With some abuse of notation, we say that  $W \in \mathcal{S}$  if every row  $\mathbf{w}^j \in \mathcal{S}$ . Finally,  $\Gamma_j$  is the diagonal covariance matrix produced via

$$\Gamma_j^{-1} = \sum_k w_{j,k} \Lambda_k^{-1}, \quad (12)$$

where  $\Lambda_k$  is defined as a diagonal matrix formed from the  $k$ -th column of matrix  $\Lambda$ .

Although the unknown  $\mathbf{x}_j$  from each task are assumed to be independent via the above distributions, they will nonetheless become linked via the common set of hyperparameters that will subsequently be estimated from the data. Additionally, from (12) we are expressing what amounts to the  $j$ -th precision matrix as a linear combination of  $K$  diagonal precision matrix basis functions. Although we could have



equally considered a linear basis expansion with respect to covariances, we chose precisions for algorithmic reasons detailed below. Additionally, the value of  $K$  can be viewed as an upper bound on the number of clusters we can expect in our data; for all experiments we simply choose  $K = L$ , the number of tasks.

Given this likelihood and prior, the posterior distribution  $p(\mathbf{x}_j | \mathbf{y}_j; \Lambda, W)$  is also a Gaussian with mean

$$\hat{\mathbf{x}}_j = \Gamma_j \Phi_j^\top (\nu I + \Phi_j \Gamma_j \Phi_j^\top)^{-1} \mathbf{y}_j. \quad (13)$$

Thus if  $\Lambda$  and  $W$  were known, we have access to a simple closed-form estimator for  $\mathbf{x}_j$ . The most challenging responsibility then becomes estimating these unknown hyperparameters. The empirical Bayesian solution to this problem is to first apply hyperpriors to  $\Lambda$  and  $W$ , integrate out the unknown  $X$ , and then compute MAP estimates via

$$\max_{\Lambda > 0, W \in \mathcal{S}} \int p(Y|X)p(X; \Lambda, W)p(\Lambda)p(W)dX. \quad (14)$$

For the covariance bases we simply assume a flat hyperprior  $p(\Lambda) = 1$ . In contrast, we assume  $p(W) \propto \exp[-1/2 \sum_{j,k} f(w_{j,k})]$ , where  $f$  is a function designed to promote a clustering effect as will be described shortly. Given the above, applying a  $-2$  log transformation to (14) produces the equivalent problem

$$\min_{\Lambda > 0, W \in \mathcal{S}} \sum_j \left[ \mathbf{y}_j \Sigma_{y_j}^{-1} \mathbf{y}_j + \log |\Sigma_{y_j}| \right] + \sum_{j,k} f(w_{j,k}), \quad (15)$$

where

$$\Sigma_{y_j} \triangleq \nu I + \Phi_j \Gamma_j \Phi_j^\top.$$

To facilitate later optimization, it will help to modify the log-det term in (15) as follows. First, using standard determinant identities we have

$$\log |\Sigma_{y_j}| \equiv T1 + T2 \quad (16)$$

$$\triangleq \log \left| \sum_k w_{j,k} \Lambda_k^{-1} + \frac{1}{\nu} \Phi_j^\top \Phi_j \right| - \log \left| \sum_k w_{j,k} \Lambda_k^{-1} \right|,$$

where irrelevant constants have been omitted. Using the fact that  $\log |\cdot|$  is a concave function in the space of positive definite, symmetric matrices,  $W \in \mathcal{S}$ , and Jensen's inequality, it follows that

$$\sum_k w_{j,k} \log |\Lambda_k| \geq - \log \left| \sum_k w_{j,k} \Lambda_k^{-1} \right|. \quad (17)$$

This upper bound has the appeal that it is linear in elements of  $W$  which will facilitate the derivation of update rules to be presented shortly. We will henceforth be concerned with minimizing the new objective function

$$\mathcal{L}(\Lambda, W) \triangleq \sum_j \left[ \mathbf{y}_j \Sigma_{y_j}^{-1} \mathbf{y}_j \right] + \sum_{j,k} f(w_{j,k}) \quad (18)$$

$$+ \sum_j \log \left| \sum_k w_{j,k} \Lambda_k^{-1} + \frac{1}{\nu} \Phi_j^\top \Phi_j \right| + \sum_{j,k} w_{j,k} \log |\Lambda_k|.$$

**Sparsity Promotion:** The log-det term in the original cost (15) is a concave, non-decreasing function of each  $\Gamma_j$ , and hence it favors sparse diagonal elements, which in turn produces a sparse  $\mathbf{x}_j$  estimate via the left multiplication in (13). But this sparsity can only be achieved if diagonal elements of the embedded basis functions  $\Lambda_k$  also converge to zero.<sup>2</sup> By virtue of the basis expansion (12) in terms of precisions, this then implies that the sparsity profile or support of  $\Gamma_j$  will mirror the sparsity profile of the *intersection* of all  $\Lambda_k$  associated with nonzero coefficients  $w_{j,k}$ . Typically this will encourage only a single unique basis function to be active for a given task  $j$ .

Note that the cost function modification using Jensen's inequality above does not interfere with this sparsity promotion agency. In fact, the upper bound gap has a minimal value of zero when either  $\mathbf{w}^j$  equals an indicator vector (all zeros and a single one), or when all  $\Lambda_k$  are equal to one another. The former will lead to a maximal  $\ell_0$  norm solution, the latter a maximal row-sparse solution.

**Cluster Promotion:** We now turn to the related clustering issues and the role of  $f$ . For this purpose, it is instructive to elucidate exactly what we mean by a cluster. We define a cluster as a group of tasks that share a common diagonal support for  $\Gamma_j$ . Without  $f$ , it is easy to show that for any value of  $\nu$ , if  $K = L$  the globally optimal solution to (18) will involve the  $k$ -th column of  $W$ ,  $\mathbf{w}_k$ , equal to a unique indicator vector for all  $k$ , each  $\Gamma_j$  will then be represented with a unique  $\Lambda_k$ , and no clustering will occur at all. In fact, there will be no clustering effect for either the purpose of avoiding local minima when  $\nu$  is small, nor for improving the effective SNR when  $\nu$  is large.

To mitigate this effect,  $f$  can be chosen to encourage  $W$  to have columns with multiple nonzero values, which is tantamount to requiring that groups of tasks must share one or more  $\Lambda_k$  basis matrices. However, because the support of any  $\Gamma_j$  will be the intersection of  $\Lambda_k$  supports associated with  $w_{j,k} > 0$ , these tasks will either share only a single  $\Lambda_k$ , or alternatively multiple different  $\Lambda_k$  will converge to the same basis matrix (or at least one with a common support). In either case, the net effect is hard clustering, where each task  $j \in \Omega_k$  will be assigned some effective  $\Lambda_k$ , and the total number of unique such basis matrices will be some  $\tilde{K} < L$ . Additionally, within each such cluster, it can be shown by extending the analysis in (Wipf et al., 2011) that the net effect on the final estimation step is as if there were

<sup>2</sup>While technically division by zero is undefined, we can still accommodate (12) and all attendant update rule derivations by considering the appropriate limiting cases along with judicious use of the Matrix Inversion Lemma and the Moore-Penrose Pseudoinverse in place of direct inverses.

an explicit, concave and nondecreasing penalty on the  $\ell_2$  row norms of  $\hat{X}_{\Omega_k}$ , which naturally favors row-sparsity.

For these reasons we choose

$$f(w) = \beta w \log w, \quad (19)$$

where  $\beta > 0$  is a constant. This  $f$  is convex over the domain  $[0, 1]$  and has a minimal value between zero and one, and therefore favors either sharing of basis functions along columns of  $W$  or merging different  $\Lambda_k$  values together via the mechanism outlined above. Importantly, many elements of  $W$  will still be pushed to exactly zero to shut off basis matrices from other clusters, provably so in certain circumstances although space here prevents a detailed treatment (Section 6 does provide empirical evidence for this however). While certainly other selections for  $f$  could potentially be more effective, this simple choice serves our purposes sufficiently well and leads to convenient update rules.

## 4 ALGORITHM DERIVATION

Optimization of (18) will involve expanding a majorization-minimization scheme suggested in (Wipf & Nagarajan, 2010) for single-task sparse estimation, where auxiliary variables are introduced to upper bound various terms in the objective function. First we use the bound

$$\frac{1}{\nu} \|\mathbf{y}_j - \Phi_j \mathbf{x}_j\|_2^2 + \mathbf{x}_j^\top \Gamma_j^{-1} \mathbf{x}_j \geq \mathbf{y}_j^\top \Sigma_{y_j}^{-1} \mathbf{y}_j \quad (20)$$

for all  $\mathbf{x}_j$ , with equality iff  $\mathbf{x}_j$  is given by (13). Now define  $\mathbf{a}_j$  as a vector formed from the diagonal of  $\sum_k w_{j,k} \Lambda_k^{-1}$ . Because the term  $T1$  in (16) is a concave, non-decreasing function of  $\mathbf{a}_j$ , we define  $h^*(\mathbf{z})$  as the concave conjugate function (Boyd & Vandenberghe, 2004) of  $h(\mathbf{a}_j) = \log \left| \sum_k w_{j,k} \Lambda_k^{-1} + \frac{1}{\nu} \Phi_j^\top \Phi_j \right|$  defined as

$$h^*(\mathbf{z}_j) \triangleq \inf_{\mathbf{a}_j} (\mathbf{z}_j^\top \mathbf{a}_j - h(\mathbf{a}_j)). \quad (21)$$

By construction we may then upper bound  $T1$  via

$$\mathbf{z}_j^\top \mathbf{a}_j - h^*(\mathbf{z}_j) \geq \log \left| \sum_k w_{j,k} \Lambda_k^{-1} + \frac{1}{\nu} \Phi_j^\top \Phi_j \right| \quad (22)$$

for all  $\mathbf{z}_j \geq 0$ , with equality iff  $\mathbf{z}_j$  is the gradient of  $T1$  with respect to  $\mathbf{a}_j$ . This can be computed in closed form using

$$\mathbf{z}_j = \nabla_{\mathbf{a}_j} (T1) = \text{diag} \left[ \left( \sum_k w_{j,k} \Lambda_k^{-1} + \frac{1}{\nu} \Phi_j^\top \Phi_j \right)^{-1} \right]. \quad (23)$$

With these upper bounds fixed, we can then optimize over  $\Lambda$  and  $W$ . First, with  $W$  fixed, optimization over  $\Lambda$  decouples and we may consider each  $\lambda_{i,k}$  individually. Collecting relevant terms we have

$$\min_{\lambda_{i,k} > 0} \sum_j \frac{w_{j,k}}{\lambda_{i,k}} (x_{i,j}^2 + z_{i,j}) + w_{j,k} \log \lambda_{i,k}. \quad (24)$$

Computing derivatives, equating to zero, and checking first-order optimality conditions we arrive at the optimal solution

$$\lambda_{i,k}^{opt} = \frac{\sum_j w_{j,k} (x_{i,j}^2 + z_{i,j})}{\sum_j w_{j,k}}, \quad \forall i, k. \quad (25)$$

Finally we fix  $\Lambda$  and optimize over  $W$ , solving separately for each row  $\mathbf{w}^j$  via

$$\min_{\mathbf{w}^j \in \mathcal{S}} \sum_{i,k} w_{j,k} \left( \frac{x_{i,j}^2 + z_{i,j}}{\lambda_{i,k}} \right) + w_{j,k} \log \lambda_{i,k} + \beta w_{j,k} \log w_{j,k}. \quad (26)$$

There exist many strategies to perform the requisite convex optimization over  $\mathbf{w}^j$ . Since (26) can be computed in closed form without the constraint  $\sum_k w_{j,k} = 1$ , we simply solve without the constraint and then normalize the resulting solution, which is a form of projected gradient method. In our experiments we found this procedure to be adequate for obtaining good results, but certainly a more precise alternative could be substituted for this step. Additionally, although these updates can be implemented such that each step is guaranteed to reduce or leave (18) unchanged, this alone is insufficient to guarantee formal convergence to a stationary point. The latter requires, for example, that the additional conditions of Zangwill's Global Convergence Theorem hold (Zangwill, 1969). However, we have not encountered any convergence issues in practice.

We refer to the aggregation of these update rules as a clustered sparse Bayesian learning (C-SBL) algorithm. The basic algorithm flow-chart/summary can be found in the supplementary file. Finally, there are only two parameters to set when using C-SBL, specifically  $\nu$  and  $\beta$ . The former can actually be learned from the data using an update rule originally proposed in (Tipping, 2001). In contrast, for  $\beta$  we adopt a simple heuristic to balance this value according to problem size. For all the simulations reported in Section 6,  $\nu$  was learned and  $\beta$  was set using this fixed rule without any additional tuning as the problem settings change.

## 5 ANALYSIS

**Low-Noise Cost Function Behavior:** We now analyze some of the properties of the underlying C-SBL cost function from (18) that make it especially suitable for the clustered sparse estimation problem. First we examine the limiting case  $\nu \rightarrow 0$ , mirroring some of our observations from Section 2, where we discussed connections with maximally sparse solutions. We also define  $\text{spark}[\Phi]$  as the smallest number of linearly dependent columns in some matrix  $\Phi$  (Donoho and Elad, 2004). In this regard we have the following:

**Theorem 1.** Assume that an optimal solution  $X^*$  to (6) exists with  $\|\mathbf{x}_j^*\|_0 < N$  and  $\text{spark}[\Phi_j] = N + 1$  for all  $j$ . Additionally, let  $\Lambda^*, W^*$  denote any global solution

of  $\lim_{\nu \rightarrow 0} \inf_{\Lambda > 0, W \in \mathcal{S}} \mathcal{L}(\Lambda, W)$ . Then the value of (13) as  $\nu \rightarrow 0$  given by  $\Gamma_j^* \Phi_j (\Phi_j \Gamma_j^* \Phi_j^\top)^\dagger \mathbf{y}_j$ , when combined across all  $j$  with  $\Gamma_j^* = \left( \sum_k w_{j,k}^* (\Lambda_k^*)^{-1} \right)^{-1}$ , forms a globally optimal solution to (6).

This result can be proven by adapting Theorem 4 from (Wipf et al., 2011), which applies to single task compressive sensing models. Note that the spark assumption is very mild and will be satisfied almost surely by any dictionary constructed via (5). Therefore, the C-SBL cost function clearly favors maximally sparse solutions in the low-noise regime as desired. Importantly however, while the global optimum of C-SBL may be equivalent to (6), the entire cost function landscape is not identical, and exploiting the cluster structure, and row-sparsity within clusters, can be advantageous in avoiding distracting local minima. Two important distinctions play a role in this regard.

First, the inclusion of the penalty term  $\sum_{j,k} f(w_{j,k})$ , by favoring solutions in clusters, naturally steers away from unpromising areas of the parameter space without correlation structure among tasks. Secondly, if we are able to determine the correct cluster structure, then there is a natural mechanism embedded in (18) to leverage the resulting row-sparsity to avoid local solutions, sometimes provably so. For example, assume for simplicity that  $\Phi_j = \Phi \forall j$ , meaning the same dictionary is used for all tasks. Also define the condition number of any matrix  $A$  as  $\kappa(A) = \|A^{-1}\|_2 \|A\|_2$ , where  $\|\cdot\|_2$  is the spectral norm.

Now assume that our measurements have been partitioned into  $\bar{K} \leq L$  clusters  $\Omega_k$ , where  $Y_{\Omega_k}$  are the columns of  $Y$  associated with cluster  $k$ . Such a clustering could be provided by an oracle, or alternatively can be viewed as an intermediate point during the optimization process whereby for every task  $j \in \Omega_k$ ,  $\Gamma_j = \Lambda_k$  for some unique  $\Lambda_k$ . We may then consider the remaining multi-task sparse estimation problems to estimate the corresponding maximally row-sparse  $X_{\Omega_k}^*$  within each cluster, holding the cluster assignments fixed, similar to problem (3).

In this scenario, Jensen's inequality collapses to an equality, the C-SBL cost function (18) decouples, and we may equivalently consider each cluster  $k$  as a separate subproblem to minimize

$$\mathcal{L}_k(\Lambda_k) \triangleq \text{tr} \left[ Y_{\Omega_k} Y_{\Omega_k}^\top (\Sigma_k)^{-1} \right] + |\Omega_k| \log |\Sigma_k|, \quad (27)$$

where  $\Sigma_k \triangleq \nu I + \Phi \Lambda_k \Phi^\top$ . Then we have the following:

**Theorem 2.** Let  $\text{spark}(\Phi) = N + 1$ . Also, let  $X_{\Omega_k}^*$  be a maximally row-sparse feasible solution to  $Y_{\Omega_k} = \Phi X_{\Omega_k}$  with  $D \triangleq \|X_{\Omega_k}^*\|_{\text{row-}\ell_0}$ . Define  $\bar{X}_{\Omega_k}^*$  as the associated collection of nonzero rows. Then if  $X_{\Omega_k}^*$  satisfies

$$\inf_{\Psi > 0} \kappa(\Psi \bar{X}_{\Omega_k}^* (\bar{X}_{\Omega_k}^*)^\top \Psi) < \frac{N}{D} \quad (28)$$

with  $\Psi \in \mathbb{R}^{D \times D}$  diagonal, then  $\lim_{\nu \rightarrow 0} \inf_{\Lambda_k > 0} \mathcal{L}_k(\Lambda_k)$  has a unique local minimum (or stationary point)  $\Lambda_k^*$ , and this point will satisfy  $\Lambda_k^* \Phi^\top (\Phi \Lambda_k^* \Phi^\top)^\dagger Y_{\Omega_k} = X_{\Omega_k}^*$ .

The supplementary file contains details of the proof. Theorem 2 dictates circumstances under which we are guaranteed to recover the maximally row-sparse solution within each cluster (assuming we are given an algorithm that converges to a stationary point), meaning we are guaranteed to solve (18) without resorting to brute-force optimization of the more challenging NP-hard problem (6). Moreover, the most relevant criteria under which this occurs depends only on the conditioning of the nonzero rows in  $X_{\Omega_k}^*$ . In words, if these rows contain complementary information regarding the true sparsity profile, as evidenced by a high condition number, no locally minimizing solutions exist. A weaker related result has already been known in the information theory community, but this result adapted to the present context would require that rows of  $\bar{X}_{\Omega_k}^*$  be strictly orthogonal (Kim et al., 2012). Additionally, Theorem 2 is independent of any RIP conditions or other strong structural assumptions on  $\Phi$  typical of compressive sensing recovery results.

Note that arguably the most common strategy for promoting row-sparse solutions is to solve problems of the form

$$\min_{X_{\Omega_k}} \sum_j h(\|\mathbf{x}_{\Omega_k}^j\|_2) \quad \text{s.t. } Y_{\Omega_k} = \Phi X_{\Omega_k}, \quad (29)$$

where  $h$  is an arbitrary non-decreasing function, and  $\mathbf{x}_{\Omega_k}^j$  denotes the  $j$ -th row of  $X_{\Omega_k}$ . Interestingly though, specialized counter-examples can be used to show that, for any such  $h$  (including the selection  $h(z) = z$  that leads to the convex  $\ell_{1,2}$  mixed-norm (Obozinski et al., 2011) commonly used in compressive sensing), there will always exist a  $\Phi$  and  $Y_{\Omega_k}$ , consistent with the stipulations of Theorem 2 such that there is guaranteed to be a stationary point not equal to  $X_{\Omega_k}^*$  when solving (29). Hence the C-SBL cost function maintains an inherent advantage at the cluster level from an optimization standpoint.

**High-Noise Cost Function Behavior:** Now we briefly consider the scenario where  $\nu$  becomes large. We first observe that the data dependent term in (18) tends towards  $\sum_j \|\mathbf{y}_j\|_2^2 / \nu + O(\nu^{-1})$  as  $\nu$  increases. Likewise the remaining  $\nu$ -dependent penalty term converges as  $\log |\Gamma_j^{-1} + (1/\nu) \Phi_j^\top \Phi_j| \rightarrow \log |\sum_k w_{j,k} \Lambda_k^{-1}| + O(\nu^{-1})$ .

By Jensen's inequality, the resulting combined factor

$$\sum_{j,k} w_{j,k} \log |\Lambda_k| + \sum_j \log \left| \sum_k w_{j,k} \Lambda_k^{-1} \right| \quad (30)$$

has a minimal value of zero when either  $w^j$  equals an indicator vector for all  $j$ , or when  $\Lambda_k$  equals some  $\Lambda'$  for all  $k$ . The former scenario will cause the weight penalty

$\sum_{j,k} f(w_{j,k})$  to become large. However, if all  $\Lambda_k = \Lambda'$ , then all of these penalty factors can effectively be minimized. Assuming the contribution from  $O(\nu^{-1})$  terms is small, this will then minimize the overall objective function. Moreover, with all  $\Lambda_k = \Lambda'$ , we by definition collapse to a single cluster, multi-task sparse estimation model as was motivated in Section 2 at low SNR.

## 6 EXPERIMENTS

This section provides empirical validation for the proposed C-SBL algorithm. We compare performance against the traditional convex  $\ell_1$  penalized regression estimator commonly using in compressive sensing, as well as three related sparse Bayesian algorithms that have previously been applied to similar problems. These include the original sparse Bayesian learning (SBL) (Tipping, 2001), a multiple measurement vector (MMV) extension of SBL (Ji et al., 2009), and the Dirichlet Process (DP) prior adaptation of multi-task Bayesian compressive sensing (Qi et al., 2008). The latter is arguably the closest competitor to C-SBL given its ability to learn clusters with sparse support. An additional sparse Bayesian algorithm from (Zhang & Rao, 2011) also addresses a multi-task sparse learning setting based upon related variational principles; however, this method cannot learn clusters, our central purpose, nor does code appear to be available for handling different sensing matrices  $\Phi_j$  for different tasks. Therefore we do not include comparisons here. We will begin with synthetic data simulations to demonstrate model properties followed by efforts to reconstruct image sequences from compressive measurements.

**Synthetic Data:** For the first experiment we generate data from  $\bar{K} = 5$  clusters. Within every cluster are 5 tasks each for a total of  $L = 25$  tasks. Each corresponding  $X_{\Omega_k}^*$  is generated with a random row-sparsity pattern distinct from one another, and with nonzero rows distributed as  $\bar{X}_{\Omega_k}^* \sim \mathcal{N}(0, 1)$  (i.e., each task has its own independent nonzero coefficients). The associated task-specific dictionaries are generated via  $\Phi_j \sim \mathcal{N}(0, 1/N)$ ; we set  $M = 256$ ,  $D \triangleq \|X_{\Omega_k}^*\|_{\text{row}-\ell_0} = 30$ , and the number of measurements per task  $N$  is varied. We then compute  $\mathbf{y}_j = \Phi_j \mathbf{x}_j^* \forall j$  in each instance and run the respective algorithms to compare the recovery performance, averaging across 50 trials.

Results are presented in Figure 1(a), where we display the normalized mean-squared error metric given by  $\langle \|\hat{X} - X^*\|_2^2 / \|X^*\|_2^2 \rangle$ . We observe that C-SBL has the lowest reconstruction error among all the methods. Additionally based on Lemma 1,  $X^*$  will almost surely be the globally optimal solution to (6). While it has been proven that regular SBL also has the same global optimum to (6) (Wipf et al., 2011), this algorithm is blind to any structure between tasks and therefore may become trapped at suboptimal local minima, leading to relatively poorer performance. On the other hand, C-SBL is more likely to reach the global

optima by exploiting cluster information. Interestingly, the DP algorithm, which also putatively leverages these clusters, does not perform significantly better than SBL, suggesting that it is non-trivial to optimally use the additional structure.

In contrast, with a different data generation mechanism, DP has demonstrated improvement over SBL but not C-SBL. Here we recreate a close approximation to experiments conducted in (Qi et al., 2008).<sup>3</sup> We begin with  $D = 27$  nonzero rows but with both amplitudes and supports shared across tasks. An innovations component is then added, whereby an additional 3 elements of each task are given random nonzero values, with task-specific, randomly generated support. Figures 1(b) and 1(c) display results as different parameters are varied. Indeed in this revised scenario DP does significantly outperform SBL; however, C-SBL retains its advantage over all algorithms.

Finally we consider reconstructions in the presence of noise. For this purpose we generate data in the same manner as was used to generate Figure 1(a), and fix  $N = 75$ ,  $M = 256$ , and  $D = 30$  while varying the SNR using additive Gaussian white noise. We also include an ideal oracle estimator that knows the true clusters. Results are displayed in Figure 1(d), where again C-SBL is observed to perform well, and in excess of 15dB SNR nearly matches even the oracle.

**Image Clustering and Reconstruction:** Here we consider a real-world application motivated in (Qi et al., 2008) that involves simultaneously reconstructing multiple images from different dynamic scenes using compressive measurements. In this scenario, tasks are images and each cluster represents a group of snapshots taken from a given dynamic scene that are likely to have a similar sparsity profile in the wavelet domain. Moreover, we may expect to have different cluster sizes and noise levels across snapshots, and moving objects behave like the innovations applied in producing Figures 1(b) and 1(c).

For this experiment we choose 5 dynamic scenes (5 clusters), each having  $\{5, 3, 3, 4, 4\}$  tasks respectively. Images have a resolution of  $64 \times 64$ , although the supplementary file contains higher resolution examples. Data are sampled using the 'db4' 2D wavelet transform using 4 scales as provided by Matlab. Each Gaussian sensing matrix  $\Phi_j$  is  $N = 2275 \times M = 5986$ , with iid elements generated as before. Although undoubtedly better performance could be obtained by selecting different transforms and/or applying different sampling rates to different scales, this is not our primary focus here. Overall we are merely adopting an established benchmark and inserting C-SBL into this pipeline

<sup>3</sup>Note that certain simulation specifics needed to exactly reproduce the results from (Qi et al., 2008) were missing (e.g. SNR), and we were unfortunately unable to obtain code from the authors.

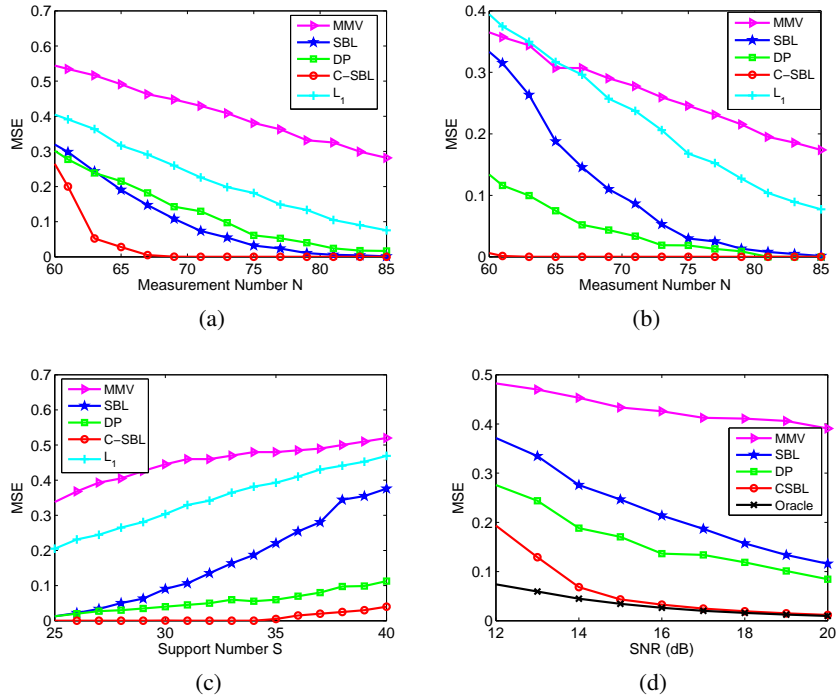


Figure 1: Synthetic data reconstruction performance comparisons; in all cases  $M = 256$ . (a) MSE versus  $N$ , with  $D = 30$ . Tasks belonging to the same cluster share the same support; nonzero coefficients are independent. (b) MSE versus  $N$ . Tasks belonging to the same cluster share the same support and coefficients over  $D = 27$  nonzero rows. Each task then has an additional 3 randomly positioned, independent nonzero elements (innovations). (c) Same as (b), only now  $N = 70$  and the total support cardinality  $S \triangleq D + 3$  is varied. (d) MSE versus SNR. Data generated as in Figure 1(b), with  $N = 75$ ,  $D = 30$ , and additive Gaussian white noise applied to achieve the desired SNR.

unaltered or specially tuned.

Figure 2 shows example reconstruction results of four out of five of the different scenes. For space consideration we only show a single reconstructed image frame from each scene cluster and compare three algorithms: C-SBL, DP, and MMV. The supplementary file contains the full results and other details. Figure 3(a) shows the normalized MSE trajectory as a function of iteration number up to convergence. In terms of both MSE (Figure 3(a)) and visual inspection (Figure 2 and supplementary), C-SBL outperforms other algorithms. In terms of per-iteration computational complexity all algorithms are approximately equal, scaling quadratically in  $M$ , and linearly in  $N$  and  $L$  with the proper implementation.

Finally, Figures 3(b) and 3(c) display the beneficial hard clustering effect of C-SBL with regard to ground truth as revealed through heat-maps of the estimated cluster matrices  $W$ . Here column permutations are irrelevant as the column labels are arbitrary. By employing C-SBL, tasks within the same group (as partitioned by the ground truth in Figure 3(b)) return nonzeros along the same columns of the estimated  $W$  (Figure 3(c)). In this way, C-SBL uses multiple bases  $\Lambda_k$  to model the clusters (different scenes in Figure 2) as evidenced by multiple nonzeros in the rows of

$W$ . However, this is the artifact of many different  $\Lambda_k$  fusing together within a true cluster, and all of these bases within a cluster must eventually share the same support (and typically magnitudes as well) by virtue of the support intersection property described in Section 3. Consequently, we can infer that C-SBL correctly learns the correct five clusters ultimately leading to the best performance (see supplementary file for DP clustering results, which fail to mirror the ground truth).

## 7 CONCLUSION

In this paper we have derived a novel Bayesian model and attendant analyses for solving multi-task sparse linear inverse problems by exploiting unknown cluster structure among the tasks. Although Bayesian models have been deployed for solving related problems, these often involve organizing postulated distributional assumptions into a complex hierarchy such that approximate inference techniques must be applied that are difficult to unpack and rationalize. In contrast, herein we rely only on a simple empirical prior and then justify this parameterization using rigorous properties of the underlying cost function that emerges. This ‘semi-Bayesian’ strategy promotes understanding of the central mechanisms at work in producing a successful algorithm, including all approximations involved, and potentially suggests targeted enhancements.

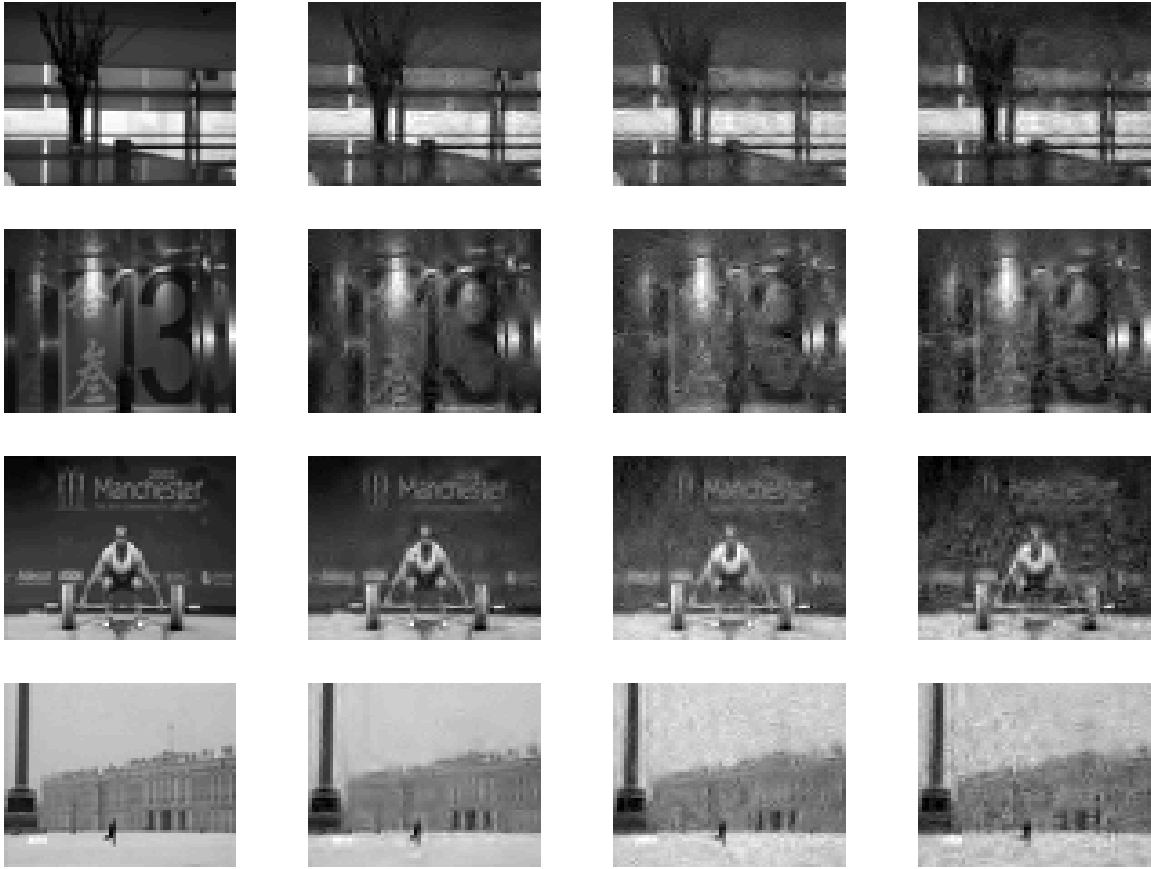


Figure 2: Reconstructions of  $64 \times 64$  images from four of the five dynamic scenes (the fifth scene would not fit owing to space considerations, but is contained in the supplementary file). From left to right: Original image, C-SBL, DP, MMV. Sampling rate is  $N/M = 0.38$ . See supplementary file for full data, higher resolution, lower sampling rate examples.

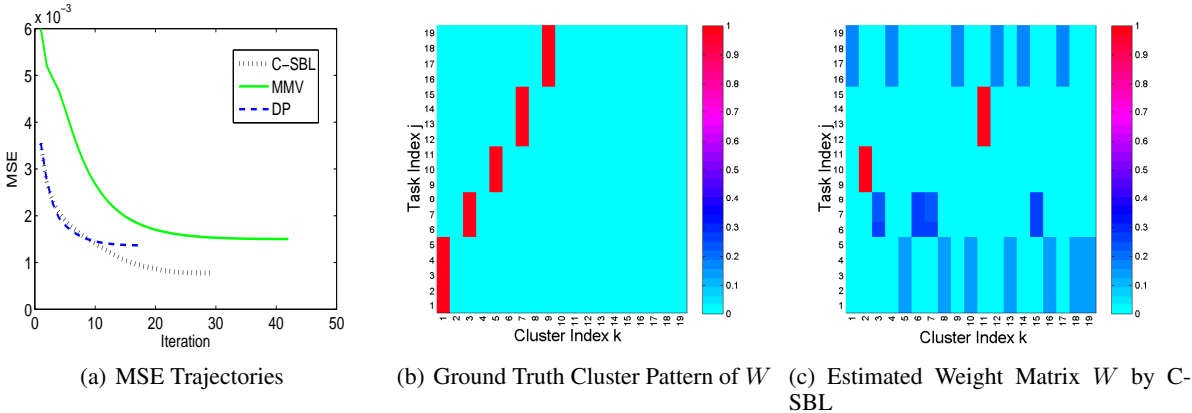


Figure 3: (a) MSE versus iteration for image reconstruction. (b) Ground truth cluster patterns (c) Estimated clustering matrix by C-SBL.

## References

- C. Archambeau, S. Guo, and O. Zoeter (2011). Sparse Bayesian multi-task learning. *Advances in Neural Information Processing Systems*, 1755-1763, Dec.
- D. Baron, M. F. Duarte, and M. B. Wakin (2009). Distributed compressive sensing. *arXiv:0901.3403v1.4729v2*.
- D. M. Blei, and M. I. Jordan (2006). Variational inference for Dirichlet process mixtures. *Bayesian Analysis*, 1(1), Mar.
- S. Boyd and L. Vandenberghe (2004). *Convex optimization*. Cambridge University Press, New York.
- V. Cevher, C. Hegde, M. F. Duarte, and R. G. Baraniuk (2008). Sparse signal recovery using markov random fields. *Advances in Neural Information Processing Systems*, 257-264, Dec.
- D. L. Donoho, and M. Elad (2003). Optimally sparse representation in general (non-orthogonal) dictionaries via  $\ell_1$  minimization. *Proceedings of The National Academy of Sciences of the United States of America*, 100(5), 2197-2202, Mar.
- E. Elhamifar, and R. Vidal (2013). Sparse subspace clustering: algorithm, theory, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11), 2765-2780, Nov
- D. Hernández-lobato, and J. M. Hernández-Lobato (2013). Learning feature selection dependencies in multi-task learning. *Advances in Neural Information Processing Systems*, 746-754, Dec.
- Z. Hong, X. Mei, D. Prokhorov, and D. Tao (2013). Tracking via robust multi-task multi-view joint sparse representation. *Computer Vision, 2013 IEEE International Conference on*, 649-656, Dec.
- A. Jalali, P. Ravikumar, and S. Sanghavi (2013). A dirty model for multiple sparse regression *IEEE Trans. Information Theory*, 59(12), 7947-7968, Dec.
- S. Ji, D. Dunson, and L. Carin (2009). Multi-task compressive sensing. *IEEE Trans. Signal Processing*, 57(1), 92-106, Jan.
- J. Kim, O. Lee, and J. Ye (2012). Compressive music: revisiting the link between compressive sensing and array signal processing. *IEEE Trans. Information Theory*, 58(1), 278-301, Jan.
- G. Obozinski, M. J. Wainwright, and M. I. Jordan (2011). Support union recovery in high-dimensional multivariate regression *The Annals of Statistics*, 39(1), 1-47, Feb.
- Y. Qi, D. Liu, D. Dunson, and L. Carin (2008). Multi-task compressive sensing with Dirichlet process priors. *Proceedings of the 25th International Conference on Machine Learning*, 768-775, July.
- N. Rao, C. Cox, R. Nowak, and T. Rogers (2013). Sparse overlapping sets lasso for multitask learning and its application to fMRI analysis. *Advances in Neural Information Processing Systems*, 2202-2210, Dec.
- T. Shi, D. Tang, L. Xu, and T. Moscibroda (2014). Correlated compressive sensing for networked data. *Conference on Uncertainty in Artificial Intelligence*, July.
- M. Soltanolkotabi, and E. J. Candès (2012). A geometric analysis of subspace clustering with outliers. *The Annals of Statistics*, 40(4), 2195-2238, July.
- M. Tipping (2001). Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1, 211-244, June.
- J. Tropp (2006). Just relax: convex programming methods for identifying sparse signals. *IEEE Trans. Information Theory*, 52(3), 1030-1051, March.
- J. Wan, Z. Zhang, J. Yan, T. Li, B. Rao, S. Fang, S. Kim, S. Risacher, A. Saykin, and L. Shen (2012). Sparse Bayesian multi-task learning for predicting cognitive outcomes from neuroimaging measures in alzheimer's disease. *Computer Vision and Pattern Recognition, 2012 IEEE Conference on*, 940-947, June.
- D. Wipf and S. Nagarajan (2010). Iterative reweighted  $\ell_1$  and  $\ell_2$  methods for finding sparse solutions. *Journal of Selected Topics in Signal Processing (Special Issue on Compressive Sensing)*, 4(2), April.
- D. Wipf, B. Rao, and S. Nagarajan (2011). Latent variable Bayesian models for promoting sparsity. *IEEE Trans. Information Theory*, 57(9), Sept.
- E. Yang, and P. D. Ravikumar (2013). Dirty statistical models. *Advances in Neural Information Processing Systems*, 611-619, Dec.
- W. Zangwill (1969). *Nonlinear programming: A unified approach*. Prentice Hall, New Jersey.
- Z. Zhang and B. Rao (2011). Sparse signal recovery with temporally correlated source vectors using sparse Bayesian learning. *IEEE Journal of Selected Topics in Signal Processing*, 5(5), 912-926, Nov.

---

# Bethe and Related Pairwise Entropy Approximations

---

Adrian Weller

Department of Engineering  
University of Cambridge  
aw665@cam.ac.uk

## Abstract

For undirected graphical models, belief propagation often performs remarkably well for approximate marginal inference, and may be viewed as a heuristic to minimize the Bethe free energy. Focusing on binary pairwise models, we demonstrate that several recent results on the Bethe approximation may be generalized to a broad family of related pairwise free energy approximations with arbitrary counting numbers. We explore approximation error and shed light on the empirical success of the Bethe approximation.

## 1 INTRODUCTION

Undirected graphical models, also called Markov random fields (MRFs), have become a central tool in machine learning, providing a powerful and compact way to describe relationships between variables. Fundamental problems are to compute the normalizing partition function, and to solve for the marginal distribution of a subset of variables (marginal inference). Both tasks are computationally intractable (Cooper, 1990), prompting great interest in approximate algorithms that perform well. One popular approach is *belief propagation* (BP, Pearl, 1988). When the underlying model topology is acyclic, this returns exact values in linear time. If the method is applied to models with cycles, termed *loopy belief propagation* (LBP), results are often strikingly good but not always, and it may not converge at all (McEliece et al., 1998).

Yedidia et al. (2001) demonstrated that fixed points of LBP correspond to stationary points of the *Bethe free energy*  $\mathcal{F}_B$  (Bethe, 1935), see §2 for definitions. Further, Heskes (2002) showed that stable fixed points correspond to local minima of the Bethe free energy. In this paper, we summarize recent results on the Bethe approximation (Welling and Teh, 2001; Weller and Jebara, 2013, 2014a,b; Weller et al., 2014), and in each case consider

how the result may be generalized by considering the broad class of pairwise entropy approximations specified by arbitrary *counting numbers*, which includes the Bethe and *tree-reweighted* approximations (TRW, Wainwright et al., 2005) as special cases. We discuss consequences and related applications, including in §5 minimizing the approximate free energy, which Weller and Jebara (2014a) recently showed, for the specific case of the Bethe approximation on attractive models, can be approximated to any  $\epsilon$ -accuracy with a *fully polynomial-time approximation scheme* (FPTAS).

In §6, we compare this family of entropy approximations to the *true* entropy, and consider how differences interact with the other form of approximation typically employed: the marginal polytope, which enforces global variable consistency, is relaxed to the local polytope, which enforces only local (pairwise) consistency. We also provide fresh insights on balanced and frustrated cycles by considering the loop series approach of Sudderth et al. (2007).

### 1.1 RELATED WORK

Related work is discussed throughout the text but here we clarify the context and contributions of our results up to §5 that build to show how to approximate the global optimum of the approximate free energy to arbitrary accuracy for general counting numbers.

**Context.** All for attractive binary pairwise models: The problem of identifying a most probable configuration (MAP inference) is solvable in polynomial-time via graph cuts (Greig et al., 1989); this generalizes to multi-label pairwise models with submodular cost functions (Schlesinger and Flach, 2006). However, aside from restricted cases (e.g. low treewidth or the *fully polynomial-time randomized approximation scheme* (FPRAS) of Jerrum and Sinclair (1993) for uniform external field), there is no way to estimate the partition function  $Z$  accurately in polynomial-time. LBP is a heuristic to find the Bethe partition function by minimizing the Bethe free energy, with  $\log Z_B = -\min \mathcal{F}_B$ , and for these mod-



els we know that  $Z_B$  is a lower bound and usually a good estimate of  $Z$  (Sudderth et al., 2007; Ruoizzi, 2012; Weller and Jebara, 2014b), but LBP may find only a local optimum or not converge at all. Various methods (e.g. CCCP, Yuille, 2002) were introduced which converge but only to a local minimum of  $\mathcal{F}_B$  with no time guarantee. Shin (2012) introduced the first polynomial-time method but this returns an approximately stationary point of the Bethe  $\mathcal{F}_B$  (i.e. a point where  $|\text{derivative of } \mathcal{F}_B| < \epsilon$ , which is useful for loop series methods, but this point may have  $\mathcal{F}_B$  value far from the global optimum; attractive not required) subject to a sparsity condition that max degree is  $O(\log n)$ . Weller and Jebara (2013) derived a PTAS for the global optimum of  $\mathcal{F}_B$  with the same sparsity condition. Weller and Jebara (2014a) improved this, providing the first FPTAS for  $\log Z_B$  for an attractive model with any topology. These applied only for the Bethe approximation.

**Contributions.** Here we broaden analysis significantly to consider any counting numbers, relying on our new Theorems 2, 5, 6 and 7, and Lemmas 3 and 4. All these extend previous results that applied only to the Bethe approximation. It is somewhat remarkable that it emerges that an attractive model admits a FPTAS for  $\log Z_A$  for any counting numbers. This is significant theoretically and will allow the benefits of non-convex free energy approximations to be explored further in future work. Theorems 2, 5 and 6 importantly apply to general (non-attractive models), as does Algorithm 1, allowing  $\log Z_A$  with any counting numbers to be computed to arbitrary accuracy, though with no polynomial-time guarantee if not attractive - still this will be useful to learn insights from small models and to benchmark accuracy of faster methods.

## 2 PRELIMINARIES

We adopt notation consistent with (Welling and Teh, 2001; Weller and Jebara, 2013, 2014a,b). Consider a binary pairwise model with  $n$  variables  $X_1, \dots, X_n \in \mathbb{B} = \{0, 1\}$  and graph topology  $(\mathcal{V}, \mathcal{E})$  with  $m = |\mathcal{E}|$  edges; that is  $\mathcal{V}$  contains nodes  $\{1, \dots, n\}$  where  $i$  corresponds to  $X_i$ , and  $\mathcal{E} \subseteq V \times V$  contains an edge for each pairwise score relationship. Let  $\mathcal{N}(i)$  be the neighbors of  $i$ . Let  $x = (x_1, \dots, x_n)$  be one particular configuration, and define its *energy*  $E(x)$  via the relationships

$$p(x) = \frac{e^{-E(x)}}{Z}, \quad E = -\sum_{i \in \mathcal{V}} \theta_i x_i - \sum_{(i,j) \in \mathcal{E}} W_{ij} x_i x_j, \quad (1)$$

where the partition function  $Z = \sum_x e^{-E(x)}$  is the normalizing constant, and  $\{\theta_i, W_{ij}\}$  specify the potentials of the model.<sup>1</sup> If  $W_{ij} \geq 0$ , the edge  $(i, j)$  is *attractive* (tending to pull its variables toward the same value); if  $W_{ij} < 0$  then it

<sup>1</sup>It is easily shown (Wainwright and Jordan, 2008) that any binary pairwise model may be reparameterized to the form in (1).

is *repulsive* (tending to push apart its variables to different values). A model is attractive iff all its edges are attractive.

### 2.1 VARIATIONAL INFERENCE AND COUNTING NUMBERS

Given any joint probability distribution  $p(X_1, \dots, X_n)$  over all variables, the Gibbs free energy is defined as  $\mathcal{F}_G(p) = \mathbb{E}_p(E) - S(p)$ , where  $S(p)$  is the (Shannon) entropy of the distribution. By considering KL divergence, it is easily shown (Wainwright and Jordan, 2008) that minimizing  $\mathcal{F}_G$  over the set of all globally valid marginals (termed the *marginal polytope*) yields a value of exactly  $-\log Z$  at the true marginal distribution, given in (1).

Since this minimization is often computationally intractable, two pairwise approximations are typically made:

1. The marginal polytope is relaxed to the *local polytope*  $\mathbb{L}$ , where only *local* consistency is required - that is we deal with a *pseudomarginal* vector  $q$ , which in our context may be considered  $\{q_i = q(X_i = 1) \forall i \in \mathcal{V}, \mu_{ij}(x_i, x_j) = q(x_i, x_j) \forall (i, j) \in \mathcal{E}\}$ , subject to constraints  $q_i = \sum_{x_j \in \mathbb{B}} \mu_{ij}(1, x_j), q_j = \sum_{x_i \in \mathbb{B}} \mu_{ij}(x_i, 1) \forall (i, j) \in \mathcal{E}$ .

The local polytope constraints imply that, given  $q_i$  and  $q_j$ ,

$$\mu_{ij} = \begin{pmatrix} 1 + \xi_{ij} - q_i - q_j & q_j - \xi_{ij} \\ q_i - \xi_{ij} & \xi_{ij} \end{pmatrix} \quad (2)$$

for some  $\xi_{ij} \in [\max(0, q_i + q_j - 1), \min(q_i, q_j)]$ .

Thus we may adopt a minimal representation with pseudomarginals specified by  $\{q_i \forall i \in \mathcal{V}\}$  singleton and  $\{\xi_{ij} \forall (i, j) \in \mathcal{E}\}$  pairwise terms.

2. The entropy  $S$  is replaced by an approximation  $S_A$  that incorporates singleton and pairwise entropy terms via *counting numbers*  $\{c_i \forall i \in \mathcal{V}, \rho_{ij} \forall (i, j) \in \mathcal{E}\}$ :

$$S_A(q) = \sum_{i \in \mathcal{V}} c_i S_i - \sum_{(i,j) \in \mathcal{E}} \rho_{ij} I_{ij}. \quad (3)$$

Here  $S_i(q_i)$  is the entropy of the singleton distribution of  $X_i$ , and  $I_{ij}(\mu_{ij})$  is the mutual information of edge  $(i, j)$  given by  $I_{ij} = S_i + S_j - S_{ij}$ , where  $S_{ij}(\mu_{ij})$  is the entropy of the pairwise distribution  $\mu_{ij}$ . Note that always  $I_{ij} \geq 0$ .<sup>2</sup>

In this paper, we shall consider the approximate partition function  $Z_A$  obtained by minimizing the corresponding approximate free energy  $\mathcal{F}_A$ , defined as follows,

$$-\log Z_A = \min_{q \in \mathbb{L}} \mathcal{F}_A(q), \quad \mathcal{F}_A(q) = \mathbb{E}_q(E) - S_A(q). \quad (4)$$

We shall also be interested in the approximate marginals given by the arg min of (4).

Eaton and Ghahramani (2013) showed that any discrete model may be arbitrarily well approximated by a binary pairwise model, though the state space may be large.

<sup>2</sup>Some instead define  $S_A = \sum_{i \in \mathcal{V}} c'_i S_i + \sum_{(i,j) \in \mathcal{E}} c'_{ij} S_{ij}$ , which is equivalent via  $c'_{ij} = \rho_{ij}, c'_i = c_i - \sum_{j \in \mathcal{N}(i)} \rho_{ij}$ .

## 2.2 CHOICE OF COUNTING NUMBERS

In the standard Bethe entropy approximation  $S_B$ , all counting numbers  $c_i$  and  $\rho_{ij}$  are set to 1. This often performs very well, yet leads to a non-convex approximate free energy  $\mathcal{F}_B$  that can be hard to optimize.

Another choice yields the well-known *tree-reweighted* approximation (TRW, Wainwright et al., 2005)  $S_T$ . Here again all  $c_i = 1$  but now the edge weights  $\rho_{ij}$  are selected from the *spanning tree polytope*, resulting in all  $\rho_{ij} \leq 1$ . Since  $I_{ij} \geq 0$ , this immediately implies that  $S_T \geq S_B$ , and hence  $Z_T \geq Z_B$ . It is also known that TRW values are bounded by true values in that  $S_T \geq S$ , hence  $Z_T \geq Z$  (whereas for many counting numbers,  $S_A$  may be above or below  $S$ , similarly  $Z_A$  may be above or below  $Z$ ; indeed, in some cases including Bethe,  $S_A$  may even be negative). We note also that  $S_T$  is concave leading to the corresponding free energy approximation  $\mathcal{F}_T$  being convex, allowing easier optimization.

Other choices of counting numbers yield a rich family of approximations, which has been studied previously. Yedidia et al. (2005) discuss counting numbers for the broader concept of *regions* which may contain any number of variables (in particular more than two). This naturally relates to *generalized belief propagation* (GBP) and associated *Kikuchi free energy approximations*. Pakzad and Anantharam (2005) and Heskes (2006) derived sufficient conditions for such free energy approximations to be convex. In this paper, we consider only pairwise counting numbers. In this context, Meshi et al. (2009) explored a wide range of pairwise counting numbers to try to find a convex free energy approximation with performance competitive to Bethe. For a subrange of models, they observed that this was possible yet still overall, Bethe performed very well. This is one of the motivations for this work, to understand better why Bethe performs so well.

Following Yedidia et al. (2005) and Meshi et al. (2009), we say that an approximation is *variable valid* if  $c_i = 1 \forall i \in \mathcal{V}$ , and is *edge valid* if  $\rho_{ij} = 1 \forall (i, j) \in \mathcal{E}$ . Their earlier work showed that variable valid approximations typically perform well compared to others, and we shall focus more attention on these models, though many of our results apply more generally to arbitrary counting numbers. Note that if all variables are independent, then variable validity is required to return the true entropy. If variables are connected in a tree, then edge validity is necessary to be exact. Bethe is unique in always being both variable and edge valid.

On a related theme, Weller et al. (2014) teased apart the two aspects of the Bethe approximation, i.e. the polytope and entropy as described in §2.1. Their results indicate that even if the optimization of (4) is performed over the marginal polytope, still the Bethe entropy approximation typically performs better than TRW. We consider polytope effects in §6.2.

## 2.3 SUBMODULARITY

A (set) function  $f : 2^X \rightarrow \mathbb{R}$  is *submodular* if  $\forall S, T \subseteq X, f(S \cap T) + f(S \cup T) \leq f(S) + f(T)$ . For finite  $X$ , this is equivalent to diminishing returns, i.e.  $\forall S \subseteq T, x \in X \setminus T, f(T \cup \{x\}) - f(T) \leq f(S \cup \{x\}) - f(S)$ .

Submodular functions have been studied extensively (Edmonds, 1970; Lovász, 1983; Bach, 2013). In some ways, they are a discrete analogue of convex functions and can be minimized efficiently. The concept can be generalized to consider any *lattice*, i.e. a partially ordered set  $(L, \preceq)$  such that  $\forall x, y \in L, \exists$  a greatest lowest bound (glb or *meet*)  $x \wedge y \in L$  and a least upper bound (lub or *join*)  $x \vee y \in L$ . A (lattice) function  $f : L \rightarrow \mathbb{R}$  is *submodular* if  $\forall x, y \in L, f(x \wedge y) + f(x \vee y) \leq f(x) + f(y)$ .

For a pairwise function  $f$  over binary variables,  $f$  is submodular iff  $f(0, 0) + f(1, 1) \leq f(0, 1) + f(1, 0)$ . It is easily shown that the energy (or cost) of an edge  $(i, j)$  is submodular iff it is attractive, i.e. iff  $W_{ij} \geq 0$ . Further, the set of vectors in  $\mathbb{R}^n$  with  $x \preceq y$  if  $x_i \leq y_i$  for all components  $i$ , is a lattice. Here  $x \wedge y$  has  $i$ th component of  $\min(x_i, y_i)$  and  $x \vee y$  has  $i$ th component of  $\max(x_i, y_i)$ .

## 2.4 FLIPPING VARIABLES

The method of *flipping* (sometimes called *switching*) binary variables will be useful for our analysis in §3.3. Given a model on variables  $\{X_i\}$ , consider a new model on  $\{X'_i\}$  where we flip a subset  $\mathcal{R}$  of the variables, i.e.  $X'_i = 1 - X_i$  for variables  $i \in \mathcal{R} \subseteq \mathcal{V}$ , and  $X'_i = X_i$  for  $i \in \mathcal{V} \setminus \mathcal{R}$ . We identify new model parameters  $\{\theta'_i, W'_{ij}\}$  as in (Weller and Jebara, 2013, §3) in order to preserve energies of all states up to a constant, hence the probability distribution over states is unchanged. If all variables are flipped (i.e.  $\mathcal{R} = \mathcal{V}$ ), new parameters are given by

$$W'_{ij} = W_{ij}, \theta'_i = -\theta_i - \sum_{j \in \mathcal{N}(i)} W_{ij}. \quad (5)$$

If the original model was attractive, so too is the new model. In general, if a subset  $\mathcal{R} \subseteq \mathcal{V}$  is flipped, let  $\mathcal{E}_t = \{\text{edges with exactly } t \text{ ends in } \mathcal{R}\}$  for  $t = 0, 1, 2$ , then we obtain

$$W'_{ij} = \begin{cases} W_{ij} & (i, j) \in \mathcal{E}_0 \cup \mathcal{E}_2, \\ -W_{ij} & (i, j) \in \mathcal{E}_1, \end{cases} \quad \theta'_i = \begin{cases} \theta_i + \sum_{(i,j) \in \mathcal{E}_1} W_{ij} & i \in \mathcal{V} \setminus \mathcal{R}, \\ -\theta_i - \sum_{(i,j) \in \mathcal{E}_2} W_{ij} & i \in \mathcal{R}. \end{cases} \quad (6)$$

The proof of the following result for general counting numbers follows the argument used by Weller and Jebara (2013) for the specific case of the Bethe approximation.

**Lemma 1.** *Flipping variables changes affected pseudo-marginal matrix entries' locations but not values. For*

any counting numbers,  $\mathcal{F}_A$  is unchanged up to a constant, hence the locations of stationary points are unaffected.

## 2.5 ATTRACTIVE AND BALANCED MODELS

A model is *attractive* iff all its edges are attractive, i.e. iff  $W_{ij} \geq 0 \forall (i, j) \in \mathcal{E}$ . As suggested by §2.3, attractive models have desirable properties, e.g. a MAP assignment may be found in polynomial time (Greig et al., 1989), and as shown in §5, we can construct a FPTAS for  $Z_A$  for any counting numbers. We remark that, as observed by Harary (1953), a general model (which may contain repulsive edges) can be mapped to an attractive model by flipping a subset of variables iff the initial model is *balanced*, that is iff it contains no *frustrated* cycles, i.e. a cycle with an odd number of repulsive edges. Hence, results that apply to attractive models may readily be extended to the wider class of balanced models.

## 3 FIRST DERIVATIVES OF $\mathcal{F}_A$

Combining (4) with (1), (2) and (3), yields

$$\begin{aligned} \mathcal{F}_A(q) = & - \sum_{i \in \mathcal{V}} \theta_i q_i - \sum_{(i,j) \in \mathcal{E}} W_{ij} \xi_{ij} \\ & - \sum_{i \in \mathcal{V}} c_i S_i + \sum_{(i,j) \in \mathcal{E}} \rho_{ij} (S_i + S_j - S_{ij}). \end{aligned} \quad (7)$$

### 3.1 OPTIMUM PAIRWISE PSEUDOMARGINALS

Differentiating (7) with respect to  $\xi_{ij}$ , we obtain

$$\begin{aligned} \frac{\partial \mathcal{F}_A}{\partial \xi_{ij}} = & -W_{ij} - \rho_{ij} \frac{\partial S_{ij}}{\partial \xi_{ij}} \\ = & -W_{ij} + \rho_{ij} \log \left[ \frac{\xi_{ij}(1 + \xi_{ij} - q_i - q_j)}{(q_i - \xi_{ij})(q_j - \xi_{ij})} \right]. \end{aligned}$$

Note that this is independent of the singleton counting numbers  $\{c_i\}$ . Welling and Teh (2001) considered the specific case of the Bethe approximation, where  $\rho_{ij} = 1$ . Solving the general case for  $\frac{\partial \mathcal{F}_A}{\partial \xi_{ij}} = 0$  leads to a quadratic equation,

$$\alpha_{ij} \xi_{ij}^2 - [1 + \alpha_{ij}(q_i + q_j)] \xi_{ij} + (1 + \alpha_{ij}) q_i q_j = 0, \quad (8)$$

where we define  $\alpha_{ij} = e^{W_{ij}/\rho_{ij}} - 1$ . Observe that here  $W_{ij}/\rho_{ij}$  plays the ‘edge count modified’ role typically performed by  $W_{ij}$  in the standard Bethe approximation. It is easily shown that (8) has just one feasible solution (Welling and Teh, 2001; Weller and Jebara, 2013), as given in the following result.

**Theorem 2.** *For general counting numbers, given singleton pseudomarginals, optimum pairwise terms (which minimize the approximate free energy) are given by*

$$\xi_{ij}^*(q_i, q_j) = \frac{1}{2\alpha_{ij}} \left( x_{ij} - \sqrt{x_{ij}^2 - 4\alpha_{ij}(1 + \alpha_{ij})q_i q_j} \right),$$

where  $\alpha_{ij} = e^{W_{ij}/\rho_{ij}} - 1$ ,  $x_{ij} = 1 + \alpha_{ij}(q_i + q_j)$ .

Henceforth we shall often consider  $\mathcal{F}_A$  as a function of just the singleton pseudomarginals  $\{q_i\}$ , with all pairwise  $\xi_{ij}$  terms being implicitly specified by their optimum values as given by Theorem 2.

As noted by Weller and Jebara (2013), (8) may be rewritten as  $\xi_{ij} - q_i q_j = \alpha_{ij}(q_i - \xi_{ij})(q_j - \xi_{ij})$ . The terms in parentheses are elements of the pairwise marginal (2), constrained to be  $\geq 0$ . By its definition,  $\alpha_{ij}$  takes the same sign as  $W_{ij}/\rho_{ij}$ , hence the following result holds.

**Lemma 3.**  $\frac{W_{ij}}{\rho_{ij}} \geq 0 \Rightarrow \xi_{ij} \geq q_i q_j$ ,  $\frac{W_{ij}}{\rho_{ij}} \leq 0 \Rightarrow \xi_{ij} \leq q_i q_j$ .

We remark that, given singleton marginals  $\{q_i\}$ , a lower edge counting number  $|\rho_{ij}|$  implies a more extreme pairwise marginal term in the sense of greater  $|\xi_{ij} - q_i q_j|$ . This is true, for example, of TRW compared to Bethe.

### 3.2 FIRST DERIVATIVES WRT $q_i$ , ASSUMING OPTIMUM PAIRWISE PSEUDOMARGINALS

We follow the approach of Welling and Teh (2001), noting that at the optimum pairwise pseudomarginals,  $\frac{\partial \mathcal{F}_A}{\partial \xi_{ij}} = 0$  for all edges, hence, holding  $q_j$  fixed  $\forall j \neq i$ ,

$$\begin{aligned} \frac{d\mathcal{F}_A}{dq_i} \Big|_{\{q_j\}} = & \frac{\partial \mathcal{F}_A}{\partial q_i} \Big|_{\{q_j, \xi_{ij}\}} + \sum_{j \in \mathcal{N}(i)} \frac{\partial \mathcal{F}_A}{\partial \xi_{ij}} \frac{\partial \xi_{ij}}{\partial q_i} \\ = & -\theta_i - c_i \frac{\partial S_i}{\partial q_i} + \sum_{j \in \mathcal{N}(i)} \rho_{ij} \frac{\partial}{\partial q_i} (S_i - S_{ij}) \\ = & -\theta_i + c_i \log \frac{q_i}{1 - q_i} \\ & + \sum_{j \in \mathcal{N}(i)} \rho_{ij} \left( -\log \frac{q_i}{1 - q_i} + \log \frac{q_i - \xi_{ij}}{1 + \xi_{ij} - q_i - q_j} \right) \\ = & -\theta_i + c_i \log \frac{q_i}{1 - q_i} + \sum_{j \in \mathcal{N}(i)} \rho_{ij} \log Q_{ij}, \end{aligned} \quad (9)$$

where as in (Weller and Jebara, 2014b), we define<sup>3</sup>

$$Q_{ij} = \left( \frac{q_i - \xi_{ij}}{1 + \xi_{ij} - q_i - q_j} \right) \left( \frac{1 - q_i}{q_i} \right). \quad (10)$$

Considering (10) and Lemma 3 yields the following.

**Lemma 4.** *If edge  $(i, j)$  is attractive, i.e.  $W_{ij} \geq 0$ , then  $\rho_{ij} \log Q_{ij} \leq 0$ .*

Gradient descent methods may be used to try to minimize  $\mathcal{F}_A$  but note these may find only a local optimum.

<sup>3</sup>Note  $Q_{ij} = \frac{\partial}{\partial q_i} (S_i - S_{ij}) = \frac{p(X_j=0|X_i=1)}{p(X_j=0|X_i=0)}$  by (2).

### 3.3 BOUNDS ON FIRST DERIVATIVES WRT $q_i$

We generalize the approach of Weller and Jebara (2014a) to bound the range of first derivatives (9) for free energy approximations with arbitrary counting numbers. An important application is the construction of an  $\epsilon$ -sufficient mesh to estimate  $\log Z_A$ , see §5.

Initially assume a model that is locally attractive around  $X_i$ , i.e.  $W_{ij} \geq 0 \forall j \in \mathcal{N}(i)$ . From (9) and Lemma 4, we obtain  $\frac{\partial \mathcal{F}_A}{\partial q_i} \leq -\theta_i + c_i \log \frac{q_i}{1-q_i}$ .

Now flip all variables, see §2.4, to consider a model with  $\{X'_i = 1 - X_i \forall i \in \mathcal{V}\}$ , keeping the same counting numbers. We obtain  $W'_{ij} = W_{ij}$  and can apply the result above to yield

$$\begin{aligned} \frac{\partial \mathcal{F}_A}{\partial q'_i} &\leq -\theta'_i + c_i \log \frac{q'_i}{1-q'_i} \\ \Leftrightarrow -\frac{\partial \mathcal{F}_A}{\partial q_i} &\leq \theta_i + W_i^+ - c_i \log \frac{q_i}{1-q_i} \quad (\text{see §2.4}), \end{aligned}$$

where we define  $W_i^+ = \sum_{j \in \mathcal{N}(i): W_{ij} \geq 0} W_{ij}$ . Combine this with the earlier result to yield a sandwich inequality,

$$-\theta_i + c_i \log \frac{q_i}{1-q_i} - W_i^+ \leq \frac{\partial \mathcal{F}_A}{\partial q_i} \leq -\theta_i + c_i \log \frac{q_i}{1-q_i}.$$

Now generalize to consider the case that  $X_i$  has some neighbors  $X_j \in \mathcal{R}$  to which it is adjacent by repulsive edges, i.e. where  $W_{ij} < 0$ . First flip just the variables in  $\mathcal{R}$ , see §2.4, and then apply the above sandwich result to yield the following Theorem, where we define the nonnegative value  $W_i^- = \sum_{j \in \mathcal{N}(i): W_{ij} \leq 0} -W_{ij}$ .

**Theorem 5.** *For arbitrary counting numbers, assuming optimum pairwise pseudomarginals, first derivatives of  $\mathcal{F}_A$  are sandwiched in the range*

$$-\theta_i + c_i \log \frac{q_i}{1-q_i} - W_i^+ \leq \frac{\partial \mathcal{F}_A}{\partial q_i} \leq -\theta_i + c_i \log \frac{q_i}{1-q_i} + W_i^-.$$

Note that both upper and lower bounds are monotonic in  $q_i$  (increasing with  $q_i$  if  $c_i > 0$ , else nonincreasing), ranging from  $-\infty$  to  $\infty$ , separated by the constant value  $W_i^- + W_i^+ = \sum_{j \in \mathcal{N}(i)} |W_{ij}|$ . See Figure 1 for an example.

## 4 SECOND DERIVATIVES OF $\mathcal{F}_A$

We extend the analysis of Weller and Jebara (2013) to derive all terms of the Hessian  $H$  for free energy approximations  $\mathcal{F}_A$  with arbitrary counting numbers.

**Theorem 6** ( $H_{ij} = \frac{\partial^2 \mathcal{F}_A}{\partial q_i \partial q_j}$  second derivatives of  $\mathcal{F}_A(q_1, \dots, q_n)$  at optimum pairwise marginals  $\xi_{ij}$ ).

$$H_{ij} = \begin{cases} \frac{q_i q_j - \xi_{ij}}{\rho_{ij} T_{ij}} & \text{if } i \neq j, (i, j) \in \mathcal{E} \\ 0 & \text{if } i \neq j, (i, j) \notin \mathcal{E} \end{cases},$$

$$H_{ii} = \frac{c_i}{q_i(1-q_i)} + \sum_{j \in \mathcal{N}(i)} \left( \frac{q_j(1-q_j)}{\rho_{ij} T_{ij}} - \frac{\rho_{ij}}{q_i(1-q_i)} \right),$$

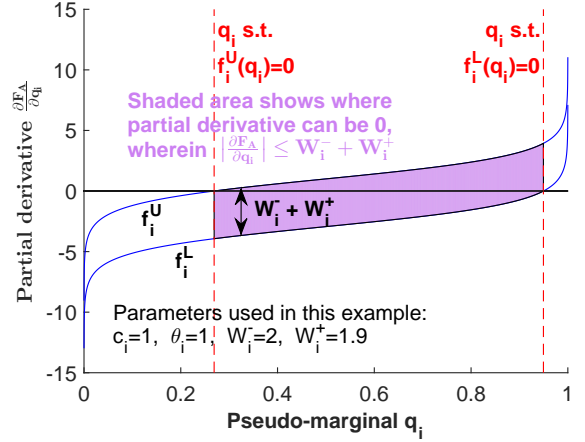


Figure 1: An example of upper and lower bounds for  $\frac{\partial \mathcal{F}_A}{\partial q_i}$ . Blue curves show monotonic upper  $f_i^U(q_i)$  and lower  $f_i^L(q_i)$  bound curves from Theorem 5, separated by constant  $W_i^- + W_i^+$ . In preprocessing, the search space is shrunk to within the dashed red lines, within which  $|\frac{\partial \mathcal{F}_A}{\partial q_i}| \leq W_i^- + W_i^+ = \sum_{j \in \mathcal{N}(i)} |W_{ij}|$ .

where  $\xi_{ij}$  takes its optimum value from Theorem 2, and  $T_{ij} = q_i q_j (1 - q_i)(1 - q_j) - (\xi_{ij} - q_i q_j)^2 \geq 0$ , with equality iff  $q_i$  or  $q_j \in \{0, 1\}$ . Proof in Appendix.

These second derivatives may be combined with the earlier gradients (9) for more efficient local minimization of  $\mathcal{F}_A$ .

## 4.1 SUBMODULARITY OF $\mathcal{F}_A$

Considering the expression for  $H_{ij}$  from Theorem 6 together with Lemma 3, observe that provided  $\rho_{ij} \neq 0$  and  $q_i, q_j \notin \{0, 1\}$ ,  $W_{ij} \geq 0 \Leftrightarrow \frac{\partial^2 \mathcal{F}_A}{\partial q_i \partial q_j} \leq 0$  (whatever the sign of  $\rho_{ij}$ ). Since third derivatives exist and are finite in this range, this yields the following result.

**Theorem 7.** *For any counting numbers with  $\rho_{ij} \neq 0 \forall (i, j) \in \mathcal{E}$ , and any discretization, an attractive model yields a submodular discrete optimization problem to estimate  $\log Z_A$ . Proof in Appendix.*

This means that considering  $\mathcal{F}_A(q_1, \dots, q_n)$  with pairwise marginals given by Theorem 2, for any discrete mesh  $\mathcal{M} = \prod_{i=1}^n M_i$ , where  $M_i$  is a finite set of points for  $q_i$  in  $[0, 1]$ , and for any counting numbers, then the discrete optimization to find the point in  $\mathcal{M}$  with lowest  $\mathcal{F}_A$  is submodular for any attractive model (hence can be solved efficiently).

## 5 OPTIMIZING THE APPROXIMATE FREE ENERGY $\mathcal{F}_A$

True marginal inference is NP-hard (Cooper, 1990), even to approximate (Dagum and Luby, 1993). However, Weller and Jebara (2014a) derived an algorithm to approximate the Bethe log-partition function,  $\log Z_B$ , to within any

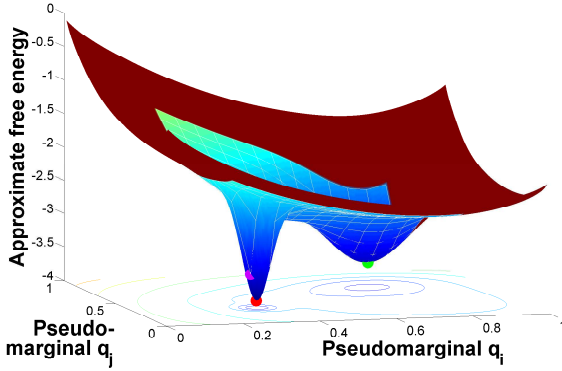


Figure 2: Stylized example for optimizing the approximate free energy over two variables. The search space is first shrunk to exclude the outer red region, then the inner blue region is discretized using an  $\epsilon$ -sufficient mesh. The red dot indicates the (continuous) global minimum. On the mesh: the purple dot has the closest location, guaranteed to have value within  $\epsilon$ , while the green dot is the lowest point, hence is the discretized optimum returned.

$\epsilon$  by constructing an  $\epsilon$ -sufficient mesh  $\mathcal{M}(\epsilon)$ , i.e. a discrete mesh over the space of singleton marginals  $[0, 1]^n$  such that the mesh point  $q^*$  with  $\min_{q \in \mathcal{M}(\epsilon)} \mathcal{F}_B(q)$  is guaranteed to have  $\mathcal{F}_B(q^*)$  within  $\epsilon$  of the global optimum of  $-\log Z_B$ . In the case of an attractive model, the discrete optimization problem was shown to be submodular, leading to a FPTAS for  $\log Z_B$ . Using Theorems 5 and 7, we extend their approach to obtain similar results for any counting numbers.

The overall mesh method is outlined in Algorithm 1 and illustrated in Figure 2. Note that we need search only over the space of singleton marginals  $[0, 1]^n$ , since pairwise terms may be computed with Theorem 2. First the search space is shrunk using the bounds of Theorem 5, since we need check only where  $\frac{\partial \mathcal{F}_A}{\partial q_i}$  can be 0. Within this range,  $|\frac{\partial \mathcal{F}_A}{\partial q_i}| \leq W_i^- + W_i^+ = \sum_{j \in \mathcal{N}(i)} |W_{ij}|$ , see Figure 1. Next, discrete mesh points for each variable's singleton marginal  $q_i$  may be selected in its range such that the step size  $\delta_i$  satisfies  $\delta_i \max |\frac{\partial \mathcal{F}_A}{\partial q_i}| \approx \frac{\epsilon}{n}$ . This ensures that, wherever the global minimum is within the space,  $\mathcal{F}_A$  cannot rise by more than  $n \frac{\epsilon}{n} = \epsilon$  at the closest mesh point. This leads to a number of mesh points in dimension  $i$  of  $N_i = O(\frac{1}{\delta_i}) = O(\frac{n}{\epsilon} \sum_{j \in \mathcal{N}(i)} |W_{ij}|)$ . If an upper bound  $W$  on edge strengths is known such that  $|W_{ij}| \leq W \forall (i, j) \in \mathcal{E}$ , then the sum of mesh points in each dimension,  $N = \sum_{i \in \mathcal{V}} N_i = O(\frac{nmW}{\epsilon})$ , where  $m = |\mathcal{E}|$ .

If the model is attractive, we obtain a FPTAS since by Theorem 7, the resulting submodular multilabel optimization problem may be solved in time  $O(N^3) = O\left(\left(\frac{nmW}{\epsilon}\right)^3\right)$  using earlier graph cut results (Schlesinger and Flach, 2006; Greig et al., 1989; Goldberg and Tarjan, 1988). If the model is balanced, then a subset of variables may be efficiently identified such that flipping them yields an attractive

---

**Algorithm 1** Mesh method to return  $\epsilon$ -approximate global optimum  $\log Z_A$  for any counting numbers.

---

**Input:**  $\epsilon$ , model parameters  $\{\theta_i, W_{ij}\}$  and counting numbers  $\{c_i, \rho_{ij}\}$

**Output:** Estimate of global optimum  $\log Z_A$  guaranteed in  $[\log Z_A - \epsilon, \log Z_A]$ , with corresponding pseudomarginals as arg for the discrete optimum

- 1: For each  $X_i$ : Compute upper and lower bound curves for  $\frac{\partial \mathcal{F}_A}{\partial q_i}$  from Theorem 5, use these to shrink the search space to a range wherein  $|\frac{\partial \mathcal{F}_A}{\partial q_i}| \leq W_i^- + W_i^+ = \sum_{j \in \mathcal{N}(i)} |W_{ij}|$ , see Figure 1.
  - 2: Construct an  $\epsilon$ -sufficient mesh as described in §5.
  - 3: Solve the resulting discrete optimization problem (efficient by Theorem 7 if the model is attractive), see §5.
- 

model (see §2.4), hence the FPTAS extends to balanced models. If the model is not balanced, there is an extensive range of methods available, see (Koller and Friedman, 2009, §13) or (Kappes et al., 2013) for recent surveys.

Various refinements to improve efficiency are discussed by Weller and Jebara (2014a) for the Bethe case. All those techniques may also be applied here, and can help significantly in practice, though they do not improve the theoretical worst case.

Other approaches to attempt to minimize the Bethe free energy have been developed (Welling and Teh, 2001; Yuille, 2002; Heskes et al., 2003; Shin, 2012), and some generalize to other counting numbers, including the message passing methods of Hazan and Shashua (2008) (guaranteed to converge for a convex free energy), Wiegerinck and Heskes (2003) or Meshi et al. (2009), but unless  $\mathcal{F}_A$  is convex, none guarantees a solution close to the global optimum.

## 6 UNDERSTANDING APPROXIMATION ERROR

We examine how the entropy approximation  $S_A$  may lead to error in the marginals, then consider other factors affecting error in the estimate of the partition function.

### 6.1 EFFECT OF APPROXIMATE ENTROPY ON MARGINALS

It has previously been observed that in cyclic graphs, there are situations where the Bethe entropy tends to pull approximate singleton marginals toward extreme values near 0 or 1, and that this tends to occur as a ‘phase transition’ in behavior when edge weights rise above some threshold (Heskes, 2004; Mooij and Kappen, 2005).<sup>4</sup> One perspec-

<sup>4</sup>Note that we describe a transition in the accuracy of approximate singleton marginals. A quite different symmetry-breaking effect is the ‘ferromagnetic-paramagnetic’ transition that relates

tive on this is algorithmic stability (Wainwright and Jordan, 2008, §7.4). A different heuristic interpretation is that it occurs as a result of LBP overcounting information when going around cycles (Ihler, 2007). Here we extend the explanatory approach of Weller et al. (2014) by considering the entropy approximation and examining the effect of different counting numbers.

To illustrate the principles, we analyze a simple model with  $n$  vertices connected such that each vertex has exactly  $d$  neighbors (such models are called  $d$ -regular), with all edge potentials symmetric of weight  $W$  and no singleton potentials (we call these models *symmetric* and *homogeneous*). Using (9), it is easily shown that, for any counting numbers, there is a stationary point of  $\mathcal{F}_A$  at a location with  $q_i = \frac{1}{2} \forall i \in \mathcal{V}$ , which by symmetry clearly also give the true singleton marginals. However, for certain counting numbers, including the Bethe parameters, when  $W$  is above a critical threshold, this stationary point is no longer a minimum, and new minima emerge that pull singleton marginals away to extreme values. The following result considers an approximation with uniform counting numbers (i.e. all  $c_i = c, \rho_{ij} = \rho$ ), and demonstrates conditions for when  $q_i = \frac{1}{2} \forall i \in \mathcal{V}$  is not a minimum, by explicitly providing a direction showing that the Hessian  $H$  is not positive semidefinite.

**Lemma 8.** *For a symmetric homogeneous  $d$ -regular model on  $n$  vertices, let  $H$  be the Hessian of the approximate free energy at  $q_i = \frac{1}{2} \forall i \in \mathcal{V}$ , using uniform counting numbers  $c_i = c \forall i \in \mathcal{V}, \rho_{ij} = \rho \forall (i, j) \in \mathcal{E}$ , then  $\mathbf{1}^T H \mathbf{1} = n \left[ 4(c - d\rho) + \frac{d}{\rho\xi} \right]$ , where  $\xi = \frac{1}{2}\sigma\left(\frac{W}{2\rho}\right)$  is the uniform optimum edge marginal term, and  $\sigma(u) = \frac{1}{1+e^{-u}}$  is the standard sigmoid function. Proof in Appendix.*

Hence,  $q_i = \frac{1}{2} \forall i$  is not a minimum if  $\omega = 4(c - d\rho) + \frac{d}{\rho\xi} < 0$ . First, note that for the Bethe approximation  $c = \rho = 1$ , and this condition reduces to  $\xi > \frac{1}{4} \frac{d}{d-1} \Leftrightarrow W > 2 \log \frac{d}{d-2}$ . Indeed, when  $W$  rises above this critical threshold, singleton marginals will move away from  $\frac{1}{2}$  (Weller et al., 2014).

In general, higher singleton counting numbers  $c$  and lower edge counting numbers  $\rho$  raise  $\omega$ , making it harder to satisfy the condition. The effect of the density of connectivity  $d$  is less clear, and depends on the other parameters. For example, consider the TRW approximation with  $c = 1$  and uniform edge weights  $\rho = \frac{2(n-1)}{nd} < 1$ , declining with  $d$ , which are optimum in this setting (Weller et al., 2014, Lemma 7), then  $\omega$  is positive and increases rapidly with  $d$  (whereas Bethe suffers in this regard by keeping  $\rho = 1$  fixed).

To understand this behavior, recall the definition of  $S_A$  in (3). As singleton counting numbers  $c_i$  rise, we add more  $S_i$  which are concave, thereby increasing convexity to the true global distribution of states (mostly aligned or not).

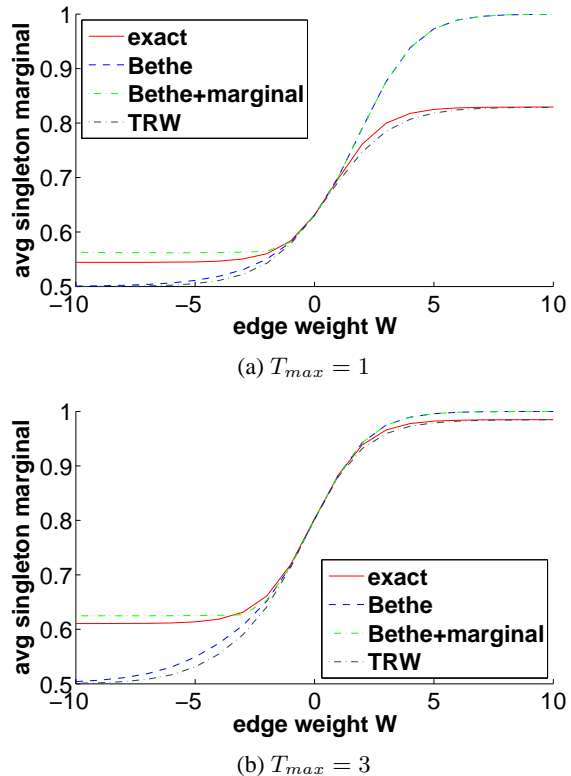


Figure 3: Average over 20 runs of singleton marginal vs. uniform symmetric edge weight  $W$  for: exact inference, Bethe approximation, Bethe+marginal polytope, and TRW (all  $\rho_{ij} = 2/3$ ). Triangle topology with random singleton potentials  $\theta_i \sim [0, T_{max}]$ . For  $W > 0$ : Bethe and Bethe+marginal overlap, exact and TRW almost overlap. For  $W < 0$  (frustrated cycle): Bethe and TRW almost overlap, as do exact and Bethe+marginal.

of  $\mathcal{F}_A$  around  $\frac{1}{2}$  and making it more likely to be a minimum. On the other hand, increasing edge terms  $\rho_{ij}$  leads to more mutual information  $I_{ij}$  being subtracted, thereby increasing concavity of  $\mathcal{F}_A$  around  $\frac{1}{2}$  and potentially pushing marginals away from  $\frac{1}{2}$ . This perspective helps to understand why a convex free energy approximation leads to algorithmic stability (Wainwright and Jordan, 2008, §7.4).

The severity of this problem for estimating singleton marginals is high when true marginals are near  $\frac{1}{2}$ , which typically occurs for small singleton potentials, but it is less problematic when true marginals are themselves near 0 or 1. The effect is illustrated in Figure 3. Note how, for positive  $W$ , the Bethe marginals are pulled toward 1 whereas TRW is almost exactly correct. The effect for  $W < 0$  is dominated instead by a polytope effect, which we discuss in the next Section.

We remark that although the entropy approximation may have a dramatic effect on the accuracy of singleton marginals, particularly for low singleton potentials (where true marginals are near  $\frac{1}{2}$ ), the effect on estimating pairwise marginals and the partition function is less clear. In-

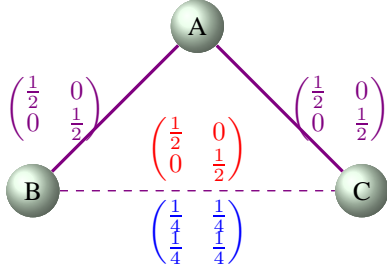


Figure 4: Illustration of the polytope effect on edge marginals. A-B and A-C are strongly coupled, B-C is very weakly coupled with all edges symmetric and attractive, and no singleton potentials. Edge marginals are shown. For B-C, above the edge (red) is the optimum in the marginal polytope (global consistency), below the edge (blue) is the optimum for the local polytope. See §6.2.

deed, Bethe typically outperforms TRW on these measures (Weller et al., 2014).

## 6.2 EFFECT OF LOCAL POLYTOPE

We revisit and expand on an example from Weller et al. (2014) to show that the impact of each of the two aspects (i.e. polytope and entropy, see §2.1) of an approximation to the partition function can pull in opposite directions. Hence, improving just the entropy approximation could lead to a *worse* approximation.

Consider the model in Figure 4, where 3 variables are connected in a triangle. Two edges are strongly attractive, and the third is very weakly attractive. The strong edge  $A - B$  ensures that  $A$  and  $B$  take the same value, similarly for  $B - C$ . Hence, in the globally consistent marginal polytope,  $B$  and  $C$  must take the same value. The global states 000 and 111 each have probability of almost  $\frac{1}{2}$ , and the pairwise marginals are shown along the edges of Figure 4. Since the model is almost a tree, we know that  $Z_B \approx Z$ . We shall examine how this arises by starting with exact inference, then switch to use the Bethe entropy approximation on the marginal polytope, and then relax the constraint set to the local polytope. We shall ignore the energy terms since they are equal here for true or approximate inference.

As noted, there are 2 states that dominate the global probability distribution, hence true  $S \approx \log 2$ . Computing the Bethe entropy on the marginal polytope, we obtain  $S_B \approx 3 \log 2 - 3 \log 2 = 0$ , which is too low by  $\log 2$ . However, when the polytope is relaxed, a better optimum is found by maximizing the edge entropy of  $B - C$  as shown under the edge in Figure 4. Since only local consistency is required, there is no longer any need for  $B$  to be equal to  $C$  and we gain the difference in edge entropy of  $2 \log 2 - \log 2 = \log 2$ , thus exactly offsetting the deficit due to Bethe entropy on the marginal polytope.

This example demonstrates that focusing exclusively on the entropy approximation, without also considering the

polytope approximation, may lead to difficulties. We highlight another aspect of the polytope approximation, in that it introduces half-integral vertices (Wainwright and Jordan, 2008). In a balanced cycle (even number of repulsive edges), this is of little consequence since the optimum energy (MAP solution) is always at an integral vertex, but in a frustrated cycle (odd number of repulsive edges, see §2.5), the energy can cause singleton marginals to be pulled towards  $\frac{1}{2}$ .<sup>5</sup> Hence, although the Bethe entropy pulls these marginals away from  $\frac{1}{2}$  on balanced cycles, the polytope effect pushes the other way on frustrated cycles, which in some cases may provide a helpful ‘balance’. Since many optimization techniques (including message passing methods) exploit the efficiencies possible with the local polytope approximation, it may in fact be desirable overall to have an entropy approximation such as Bethe, for this offsetting effect. See Figure 3 in the region  $W < 0$  for an illustration, where the Bethe+marginal optimization was performed using the Frank-Wolfe algorithm (Frank and Wolfe, 1956).

## 6.3 BOUNDS ON $Z_A$

While the TRW approximation has  $Z_T \geq Z$  by construction, until recently there were no guarantees on the performance of the Bethe approximation, though it typically yields very good results. Sudderth et al. (2007) proved that  $Z_B \leq Z$  for a range of attractive binary pairwise models, and conjectured that this bound holds for all attractive models. This was proved true by Ruozzi (2012) using the method of graph covers, and then also by Weller and Jebara (2014b) by combining the idea of clamping variables with analyzing properties of the derivatives of  $\mathcal{F}_B$ .

In this Section, we use the loop series method (Sudderth et al., 2007; Chertkov and Chernyak, 2006) to show that for certain other models, we can prove that  $Z_B \geq Z$ . For such models, this immediately implies that the Bethe approximation is better for estimating  $Z$  than any approximation with  $c_i = 1 \forall i \in \mathcal{V}$  (variable valid) and  $\rho_{ij} \leq 1 \forall (i, j) \in \mathcal{E}$  (from the definition of  $S_A$ , see §2.1-2.2). In particular, for these models,  $Z \leq Z_B \leq Z_T$ .

Sudderth et al. (2007) showed that  $Z/Z_B = 1 +$  a series of terms, one term for each *generalized loop*, which is a subgraph such that no vertex has degree 1, and demonstrated that each of the terms in the series is  $\geq 0$  for certain models, and hence  $Z_B \leq Z$  for these cases. See Appendix for background on this approach. In particular, if there is exactly one cycle in the model, then there is only one term in the series and if the cycle is attractive, then this term is positive. We note that this immediately generalizes to a cycle that is balanced (see §2.5 for definitions).

Here we apply similar analysis (Sudderth et al., 2007, §3-4, or see Appendix), and observe that if there is exactly one

<sup>5</sup>This can lead the Bethe optimum of a strongly frustrated cycle to occur at a location where  $S_B < 0$ .

cycle and it is frustrated, then the term is negative, thus proving that for such models,  $Z_B \geq Z$ .

Interestingly, Weller and Jebara (2014b) have shown that for the case of a model with one balanced cycle,  $\frac{1}{2}Z \leq Z_B \leq Z$ , so although  $Z_B$  is lower than  $Z$ , it cannot be by much even for very strong edge weights; whereas for a single frustrated cycle, there is no limit to how large  $Z_B/Z$  can rise. This suggests that for a general model, the accuracy of  $Z_B$  will depend on the blend of balanced and frustrated cycles, where in a sense frustrated cycles cause greater trouble than balanced cycles, though to understand how the effects combine in a model with multiple cycles will require further analysis. Since  $Z_B$  performs well even for attractive models (Sudderth et al., 2007), this indicates that, for estimating the partition function, practitioners should use approximations with  $\rho_{ij} < 1$  (such as TRW) with caution.

The loop series method extends to models with more than one cycle but the analysis becomes more complicated. Again using the approach of Sudderth et al. (2007), we can conclude more generally that  $Z_B \geq Z$  for any model such that every generalized loop contains an odd number of repulsive edges (this is a sort of generalized frustrated cycle), and the Bethe optimum marginals for every variable that has an odd degree  $\geq 3$  in any generalized loop, are either all  $\leq \frac{1}{2}$  or all  $\geq \frac{1}{2}$  (see Appendix).

#### 6.4 DERIVATIVES WRT COUNTING NUMBERS

We are interested in exploring which counting numbers lead to accurate inference as measured by errors in the estimates of the partition function and marginals. Considering (7) and using the envelope theorem (Milgrom, 1999, Theorem 1), we have right derivatives:

$$\begin{aligned} \frac{\partial \log Z_A}{\partial c_i} &= \max_{q \in X} S_i(q_i), \\ \frac{\partial \log Z_A}{\partial \rho_{ij}} &= \max_{q \in X} [S_{ij}(\mu_{ij}) - S_i(q_i) - S_j(q_j)], \end{aligned} \quad (11)$$

where  $X$  is the set of all  $\arg \min \mathcal{F}_A$ .<sup>6</sup> The left derivatives correspondingly take the min rather than the max of the same expressions. If the minimum of  $\mathcal{F}_A$  is unique, as is the case for any convex  $\mathcal{F}_A$ , then the right and left derivatives are equal.

For tractable models, where the exact partition function  $Z$  may be computed, this will allow exploration over the range of counting numbers that yield accurate partition functions. It will be interesting to investigate robustness

<sup>6</sup>This generalizes an earlier result for convex free energies (Meshi et al., 2009, Prop 5.2), which itself generalized a result of Wainwright et al. (2005). The envelope theorem is similar to Danskin's theorem (Bertsekas, 1995). Recall  $\log Z_A = -\min \mathcal{F}_A$ . Intuitively, for multiple  $\arg \min$  locations, each may vary at a different rate, thus for the right derivative, we must take the max of the derivative over all the locations.

of the quality of the partition function estimate to changes in model potentials, and accuracy of marginals, though this is outside the scope of the current work.

Others have investigated ways to optimize counting numbers. Wiegerinck and Heskes (2003) proposed a method using linear response theory. They also discussed alpha-divergence measures, an idea developed further by Minka (2005), who fascinatingly frames (fractional) BP and (power) EP under a general framework of iterative minimization of alpha-divergence, yielding insight into which measures may be expected to perform well for different objectives, though concluding that this is difficult to predict.

## 7 CONCLUSION

We have shown how recent results for the Bethe approximation may be extended to handle the broad range of pairwise approximations using any counting numbers. Our analysis builds on earlier work (Welling and Teh, 2001; Yedidia et al., 2005; Meshi et al., 2009; Sudderth et al., 2007; Weller and Jebara, 2013, 2014a), providing new insights and deepening our understanding of how best to perform inference in practice. This is important given the popularity of LBP and TRW approximations. Further, it provides a valuable toolbox for further exploration.

Areas for future investigation include trying to understand better how to predict which approach will work well for a given model, and analyzing the performance of message passing algorithms with different counting numbers (where our  $\epsilon$ -accurate approach provides a valuable benchmark).

#### Acknowledgements

The author thanks Ofer Meshi for fruitful discussions and for sharing code, and the anonymous reviewers for helpful comments and suggestions.

#### References

- F. Bach. Learning with submodular functions: A convex optimization perspective. *Foundations and Trends in Machine Learning*, 6(2-3):145–373, 2013.
- D. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1995.
- H. Bethe. Statistical theory of superlattices. *Proc. R. Soc. Lond. A*, 150(871):552–575, 1935.
- M. Chertkov and M. Chernyak. Loop series for discrete statistical models on graphs. *J. Stat. Mech.*, 2006.
- G. Cooper. The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 42:393–405, 1990.
- P. Dagum and M. Luby. Approximate probabilistic reasoning in Bayesian belief networks is NP-hard. *Artificial Intelligence*, 60:141–153, 1993.
- F. Eaton and Z. Ghahramani. Model reductions for inference: Generality of pairwise, binary, and planar factor graphs. *Neural Computation*, 25(5):1213–1260, 2013.



- J. Edmonds. Submodular functions, matroids, and certain polyhedra. *Edited by G. Goos, J. Hartmanis, and J. van Leeuwen*, page 11, 1970.
- M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110, 1956. ISSN 1931-9193. doi: 10.1002/nav.3800030109.
- A. Goldberg and R. Tarjan. A new approach to the maximum flow problem. *Journal of the ACM*, 35:921–940, 1988.
- D. Greig, B. Porteous, and A. Scheult. Exact maximum a posteriori estimation for binary images. *J. Royal Statistical Soc., Series B*, 51(2):271–279, 1989.
- F. Harary. On the notion of balance of a signed graph. *Michigan Mathematical Journal*, 2:143–146, 1953.
- T. Hazan and A. Shashua. Convergent message-passing algorithms for inference over general graphs with convex free energies. In *UAI*, 2008.
- T. Heskes. Stable fixed points of loopy belief propagation are minima of the Bethe free energy. In *Neural Information Processing Systems*, 2002.
- T. Heskes. On the uniqueness of loopy belief propagation fixed points. *Neural Computation*, 16(11):2379–2413, 2004.
- T. Heskes. Convexity arguments for efficient minimization of the Bethe and Kikuchi free energies. *Journal of Artificial Intelligence Research*, 26:153–190, 2006.
- T. Heskes, K. Albers, and B. Kappen. Approximate inference and constrained optimization. In *UAI*, pages 313–320, 2003.
- A. Ihler. Accuracy bounds for belief propagation. In *Uncertainty in Artificial Intelligence (UAI)*, 2007.
- M. Jerrum and A. Sinclair. Polynomial-time approximation algorithms for the Ising model. *SIAM J. Comput.*, 22(5):1087–1116, 1993.
- J. Kappes, B. Andres, F. Hamprecht, C. Schnörr, S. Nowozin, D. Batra, S. Kim, B. Kausler, J. Lellmann, N. Komodakis, and C. Rother. A comparative study of modern inference techniques for discrete energy minimization problems. In *CVPR*, 2013.
- D. Koller and N. Friedman. *Probabilistic Graphical Models - Principles and Techniques*. MIT Press, 2009.
- F. Korč, V. Kolmogorov, and C. Lampert. Approximating marginals using discrete energy minimization. Technical report, IST Austria, 2012.
- L. Lovász. Submodular functions and convexity. In A. Bachem, M. Grötschel, and B. Korte, editors, *Mathematical Programming – The State of the Art*, pages 235–257, Berlin, 1983. Springer-Verlag.
- R. McEliece, D. MacKay, and J. Cheng. Turbo decoding as an instance of Pearl’s “Belief Propagation” algorithm. *IEEE Journal on Selected Areas in Communications*, 16(2):140–152, 1998.
- O. Meshi, A. Jaimovich, A. Globerson, and N. Friedman. Convexifying the Bethe free energy. In *UAI*, pages 402–410, 2009.
- P. Milgrom. The envelope theorems. *Department of Economics, Stanford University, Mimeo*, 1999. URL <http://www-siepr.stanford.edu/workp/swp99016.pdf>.
- T. Minka. Divergence measures and message passing. *Technical Report MSR-TR-2005-173*, 2005.
- J. Mooij and H. Kappen. On the properties of the Bethe approximation and loopy belief propagation on binary networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2005.
- P. Pakzad and V. Anantharam. Belief propagation and statistical physics. In *Princeton University*, 2002.
- P. Pakzad and V. Anantharam. Estimation and marginalization using Kikuchi approximation methods. *Neural Computation*, 17(8):1836–1873, 2005.
- J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.
- N. Ruoizzi. The Bethe partition function of log-supermodular graphical models. In *Neural Information Processing Systems*, 2012.
- D. Schlesinger and B. Flach. Transforming an arbitrary minsum problem into a binary one. Technical report, Dresden University of Technology, 2006.
- J. Shin. Complexity of Bethe approximation. In *Artificial Intelligence and Statistics*, 2012.
- E. Sudderth, M. Wainwright, and A. Willsky. Loop series and Bethe variational bounds in attractive graphical models. In *NIPS*, 2007.
- M. Wainwright and M. Jordan. Graphical models, exponential families and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.
- M. Wainwright, T. Jaakkola, and A. Willsky. A new class of upper bounds on the log partition function. *IEEE Transactions on Information Theory*, 51(7):2313–2335, 2005.
- A. Weller and T. Jebara. Bethe bounds and approximating the global optimum. In *Artificial Intelligence and Statistics (AISTATS)*, 2013.
- A. Weller and T. Jebara. Approximating the Bethe partition function. In *Uncertainty in Artificial Intelligence (UAI)*, 2014a.
- A. Weller and T. Jebara. Clamping variables and approximate inference. In *Neural Information Processing Systems (NIPS)*, 2014b.
- A. Weller, K. Tang, D. Sontag, and T. Jebara. Understanding the Bethe approximation: When and how can it go wrong? In *Uncertainty in Artificial Intelligence (UAI)*, 2014.
- M. Welling and Y. Teh. Belief optimization for binary networks: A stable alternative to loopy belief propagation. In *Uncertainty in Artificial Intelligence (UAI)*, 2001.
- W. Wiegand and T. Heskes. Fractional belief propagation. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 438–445. MIT Press, 2003.
- J. Yedidia, W. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. In *International Joint Conference on Artificial Intelligence, Distinguished Lecture Track*, 2001.
- J. Yedidia, W. Freeman, and Y. Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Trans. Information Theory*, pages 2282–2312, 2005.
- A. Yuille. CCCP algorithms to minimize the Bethe and Kikuchi free energies: Convergent alternatives to belief propagation. *Neural Computation*, 14:1691–1722, 2002.

---

# Efficient Transition Probability Computation for Continuous-Time Branching Processes via Compressed Sensing

---

**Jason Xu**

Department of Statistics  
University of Washington  
Seattle, WA 98195

**Vladimir N. Minin**

Departments of Statistics and Biology  
University of Washington  
Seattle, WA 98195

## Abstract

Branching processes are a class of continuous-time Markov chains (CTMCs) with ubiquitous applications. A general difficulty in statistical inference under partially observed CTMC models arises in computing transition probabilities when the discrete state space is large or uncountable. Classical methods such as matrix exponentiation are infeasible for large or countably infinite state spaces, and sampling-based alternatives are computationally intensive, requiring integration over all possible hidden events. Recent work has successfully applied generating function techniques to computing transition probabilities for linear multi-type branching processes. While these techniques often require significantly fewer computations than matrix exponentiation, they also become prohibitive in applications with large populations. We propose a compressed sensing framework that significantly accelerates the generating function method, decreasing computational cost up to a logarithmic factor by only assuming the probability mass of transitions is sparse. We demonstrate accurate and efficient transition probability computations in branching process models for blood cell formation and evolution of self-replicating transposable elements in bacterial genomes.

## 1 INTRODUCTION

Continuous-time branching processes are widely used in stochastic modeling of population dynamics, with applications including cell biology, genetics, epidemiology, quantum optics, and nuclear fission [Renshaw, 2011]. With the exception of the well-studied class of birth-death processes, which have known expressions for many quantities relevant to probabilistic inference [Crawford et al., 2014], branching processes pose significant inferential challenges. In

particular, closed forms for finite-time *transition probabilities*, the conditional probability that a trajectory ends at a given state, given a starting state and time interval, are unavailable. These transition probabilities are crucial in many inferential approaches, comprising the observed likelihood function when data from the process are available at a set of discrete times. The likelihood function is of central importance in frequentist and Bayesian methods, and any statistical framework involving observed data likelihood evaluation requires transition probability computations. Without the ability to fully leverage the branching structure, studies must rely on general CTMC estimation techniques or model approximations [Rosenberg et al., 2003, Golinelli et al., 2006, El-Hay et al., 2006].

Computation of transition probabilities is the usual bottleneck in model-based inference using CTMCs [Hajiaghayi et al., 2014], requiring a marginalization over the infinite set of possible end-point conditioned paths. Classically, this marginalization is accomplished by computing the matrix exponential of the infinitesimal generator of the CTMC. However, this procedure has cubic runtime complexity in the size of the state space, becoming prohibitive even for state spaces of moderate sizes. Alternatives also have their shortcomings: *uniformization* methods use a discrete-time “skeleton” chain to approximate the CTMC but rely on a restrictive assumption that there is a uniform bound on all rates [Grassmann, 1977, Ross, 1987, Rao and Teh, 2011]. Typically, practitioners resort to sampling-based approaches via Markov chain Monte Carlo (MCMC). Specifically, particle-based methods such as sequential Monte Carlo (SMC) and particle MCMC [Doucet et al., 2000, Andrieu et al., 2010] offer a complementary approach whose runtime depends on the number of imputed transitions rather than the size of the state space. However, these SMC methods have several limitations—in many applications, a prohibitively large number of particles is required to impute waiting times and events between transitions, and degeneracy issues are a common occurrence, especially in longer time series. A method by Hajiaghayi et al. [2014] accelerates particle-based methods by marginalizing holding times analytically, but has cubic

runtime complexity in the number of imputed jumps between observations and is recommended for applications with fewer than one thousand events occurring between observations.

Recent work by Xu et al. [2014] has extended techniques for computing transition probabilities in birth-death models to linear multi-type branching processes. This approach involves expanding the probability generating function (PGF) of the process as a Fourier series, and applying a Riemann sum approximation to its inversion formula. This technique has been used to compute numerical transition probabilities within a maximum likelihood estimation (MLE) framework, and has also been applied within Expectation Maximization (EM) algorithms [Doss et al., 2013, Xu et al., 2014]. While this method provides a powerful alternative to simulation and avoids costly matrix operations, the Riemann approximation to the Fourier inversion formula requires  $\mathcal{O}(N^b)$  PGF evaluations, where  $b$  is the number of particle types and  $N$  is the largest population size at endpoints of desired transition probabilities. This complexity is no worse than linear in the size of the state space, but can also be restrictive: a two-type process in which each population can take values in the thousands would require millions of PGF evaluations to produce transition probabilities over an observation interval. This can amount to hours of computation in standard computing architectures, because evaluating PGFs for multi-type branching processes involves numerically solving systems of ordinary differential equations (ODEs). Such computations become infeasible within iterative algorithms.

In this paper, we focus our attention on the efficient computation of transition probabilities in the presence of sparsity, presenting a novel compressed sensing framework that dramatically reduces the computational cost of inverting the PGF. We apply our compressed sensing generating function (CSGF) algorithm to a branching process model used to study hematopoiesis — a process of blood cell formation — as well as a birth-death-shift process with applications to molecular epidemiology, and see that the sparsity assumption is valid for scientifically realistic rates of the processes obtained in previous statistical studies. We compare performance of CSGF to transition probability computations without taking advantage of sparsity, demonstrating a high degree of accuracy while achieving significant improvements in runtime.

## 2 MARKOV BRANCHING PROCESSES

A branching process is a Markov process in which a collection of *independently acting* individuals, or particles, can reproduce and die according to a probability distribution. We consider continuous-time, multi-type branching processes that take values over a discrete state space. In this setting, each particle type can have a distinct mean lifespan

and reproductive probabilities, and lives for an exponentially distributed length of time. At time of death, a particle can give rise to particles of its own type as well as other types.

Denote a linear, multi-type branching process by the random vector  $\mathbf{X}(t)$  taking values in a discrete state space  $\Omega$ , with  $X_i(t)$  denoting the number of type  $i$  particles present at time  $t \geq 0$ . For exposition and notational simplicity, we will focus on the two-type case. Each type  $i$  particle produces  $k$  type 1 particles and  $l$  type 2 particles with *instantaneous rates*  $a_j(k, l)$  upon completion of its lifespan, and the rates of no event occurring are defined as

$$\alpha_1 := a_1(1, 0) = - \sum_{(k,l) \neq (1,0)} a_1(k, l)$$

$$\alpha_2 := a_2(0, 1) = - \sum_{(k,l) \neq (0,1)} a_2(k, l),$$

so that  $\sum_{k,l} a_i(k, l) = 0$  for  $i = 1, 2$ . The linearity assumption implies that overall rates are multiplicative in the number of particles. For example, the infinitesimal probability of jumping to  $k$  type 1 and  $l$  type 2 particles beginning with  $j$  type 1 particles over a short interval of time  $h$  is

$$\Pr \{ \mathbf{X}(h) = (k, l) | \mathbf{X}(0) = (j, 0) \} = j \cdot a_1(k, l) \cdot h + o(h).$$

Subsequently, offspring of each particle evolve according to the same set of instantaneous rates, and these rates  $a_j(k, l)$  do not depend on  $t$  so that the process is *time-homogeneous*. Together these assumptions imply that each type  $i$  particle has exponentially distributed lifespan with rate  $-\alpha_i$ , and  $\mathbf{X}(t)$  evolves over time as a CTMC [Guttorp, 1995].

### 2.1 Transition probabilities

Dynamics of a CTMC are determined by its transition function

$$p_{\mathbf{x}, \mathbf{y}}(t) = \Pr(\mathbf{X}(t+s) = \mathbf{y} | \mathbf{X}(s) = \mathbf{x}), \quad (1)$$

where time-homogeneity implies independence of the value of  $s$  on the right hand side. When the state space  $\Omega$  is small, one can exponentiate the  $|\Omega|$  by  $|\Omega|$  *infinitesimal generator* or rate matrix  $\mathbf{Q} = \{q_{\mathbf{x}, \mathbf{y}}\}_{\mathbf{x}, \mathbf{y} \in \Omega}$ , where the entries  $q_{\mathbf{x}, \mathbf{y}}$  denote the instantaneous rates of jumping from state  $\mathbf{x}$  to  $\mathbf{y}$ , to compute transition probabilities:

$$\mathbf{P}(t) := \{p_{\mathbf{x}, \mathbf{y}}(t)\}_{\mathbf{x}, \mathbf{y} \in \Omega} = e^{\mathbf{Q}t} = \sum_{k=0}^{\infty} \frac{(\mathbf{Q}t)^k}{k!}. \quad (2)$$

These transition probabilities are fundamental quantities in statistical inference for data generated from CTMCs. For instance, if  $\mathbf{X}(t)$  is observed at times  $t_1, \dots, t_J$  and  $\mathbf{D}$  represents the 2 by  $J$  matrix containing the observed data, the *observed data log-likelihood* is given by

$$\ell_o(\mathbf{D}; \boldsymbol{\theta}) = \sum_{j=1}^{J-1} \log p_{\mathbf{X}(t_j), \mathbf{X}(t_{j+1})}(t_{j+1} - t_j; \boldsymbol{\theta}), \quad (3)$$

where the vector  $\theta$  parametrizes the rates  $a_j(k, l)$ . Maximum likelihood inference that seeks to find the value  $\hat{\theta}$  that optimizes (3) as well as Bayesian methods where likelihood calculations arise in working with the posterior density (up to a proportionality constant) fundamentally rely on the ability to calculate transition probabilities. Having established their importance in probabilistic inference, we focus our discussion in this paper on computing these transition probabilities in a continuous-time branching process.

## 2.2 Generating function methods

Matrix exponentiation is cubic in  $|\Omega|$  and thus prohibitive in many applications, but we may take an alternate approach by exploiting properties of the branching process. Xu et al. [2014] extend a generating function technique used to compute transition probabilities in birth-death processes to the multi-type branching process setting. The probability generating function (PGF) for a two-type process is defined

$$\begin{aligned} \phi_{jk}(t, s_1, s_2; \theta) &= \mathbb{E}_{\theta} \left( s_1^{X_1(t)} s_2^{X_2(t)} \mid X_1(0) = j, X_2(0) = k \right) \\ &= \sum_{l=0}^{\infty} \sum_{m=0}^{\infty} p_{(jk),(lm)}(t; \theta) s_1^l s_2^m; \end{aligned} \quad (4)$$

this definition extends analogously for any  $m$ -type process. We suppress dependence on  $\theta$  for notational convenience. Bailey [1964] provides a general technique to derive a system of differential equations governing  $\phi_{jk}$  using the Kolmogorov forward or backward equations given the instantaneous rates  $a_j(k, l)$ . It is often possible to solve these systems analytically for  $\phi_{jk}$ , and even when closed forms are unavailable, numerical solutions can be efficiently obtained using standard algorithms such as Runge-Kutta methods [Butcher, 1987].

With  $\phi_{jk}$  available, transition probabilities are related to the PGF (4) via differentiation:

$$p_{(jk),(lm)}(t) = \frac{1}{l!} \frac{1}{m!} \frac{\partial^l}{\partial s_1^l} \frac{\partial^m}{\partial s_2^m} \phi_{jk}(t) \Big|_{s_1=s_2=0}. \quad (5)$$

This repeated differentiation is computationally intensive and numerically unstable for large  $l, m$ , but following Lange [1982], we can map the domain  $s_1, s_2 \in [0, 1] \times [0, 1]$  to the boundary of the complex unit circle, setting  $s_1 = e^{2\pi i w_1}, s_2 = e^{2\pi i w_2}$ . The generating function becomes a Fourier series whose coefficients are the desired transition probabilities

$$\phi_{jk}(t, e^{2\pi i w_1}, e^{2\pi i w_2}) = \sum_{l,m=0}^{\infty} p_{(jk),(lm)}(t) e^{2\pi i l w_1} e^{2\pi i m w_2}.$$

Applying a Riemann sum approximation to the Fourier inversion formula, we can now compute the transition prob-

abilities via integration instead of differentiation:

$$\begin{aligned} p_{(jk),(lm)}(t) &= \int_0^1 \int_0^1 \phi_{jk}(t, e^{2\pi i w_1}, e^{2\pi i w_2}) e^{-2\pi i l w_1} \\ &\quad \times e^{-2\pi i m w_2} dw_1 dw_2 \\ &\approx \frac{1}{N^2} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} \phi_{jk} \left( t, e^{2\pi i u/N}, e^{2\pi i v/N} \right) \\ &\quad \times e^{-2\pi i l u/N} e^{-2\pi i m v/N}. \end{aligned} \quad (6)$$

In practice, the set of transition probabilities  $S = \{p_{(jk),(lm)}(t)\}$  for all  $l, m = 0, \dots, N$ , given initial values of  $(j, k)$ , can be obtained via the Fast Fourier Transform (FFT), described in Section 4. It is necessary to choose  $N > l, m$ , since exponentiating the roots of unity can yield at most  $N$  distinct values:

$$e^{-2\pi i m v/N} = e^{-2\pi i (mv \bmod N)/N}.$$

This is related to the Shannon-Nyquist criterion [Shannon, 2001], which dictates that the number of samples required to recover a signal must match its highest frequency. Thus, calculating “high frequency” coefficients—when  $l, m$  take large values—requires  $\mathcal{O}(N^2)$  numerical ODE solutions, which becomes computationally expensive for large  $N$ .

**Sparsity:** Given an initial state  $\mathbf{X}(0) = (j, k)$ , the support of transition probabilities is often concentrated over a small range of  $(l, m)$  values. For example, if  $\mathbf{X}(t) = (800, 800)$ , then the probability that the entire process becomes extinct,  $\mathbf{X}(t+s) = (0, 0)$ , is effectively zero unless particle death rates are very high or  $s$  is a very long time interval. In many realistic applications,  $p_{(800,800),(l,m)}(s)$  has non-negligible mass on a small support, for instance only over  $l, m$  values between 770 and 820. While their values can be computed using Equation (6) for a choice of  $N > 820$ , requiring  $N^2$  ODE evaluations toward computing only  $(820 - 770)^2$  nonzero probabilities seems wasteful. Similarly, in an example with high birth rates but low death rates, probabilities  $p_{(800,800),(l,m)}(s)$  may be concentrated around some mean values of  $(l, m)$  much larger than  $(j, k)$ , and other processes may feature concentration of probability mass in one or several modes. While sparsity is not always available—for instance, support may be spread out in applications when observation times are very far apart—a general sparsity assumption is very reasonable when data are observed relatively frequently relative to the branching process rates. To exploit the sparsity in such settings, we bridge aforementioned branching process techniques to the literature of *compressed sensing*.

## 3 COMPRESSED SENSING

Originally developed in an information theoretic setting, the principle of compressed sensing (CS) states that an un-

known sparse signal can be recovered accurately and often perfectly from significantly fewer samples than dictated by the Shannon-Nyquist rate, at the cost of solving a convex optimization problem [Donoho, 2006, Candès, 2006]. CS is a robust tool to collect high-dimensional sparse data from a low-dimensional set of measurements and has been applied to a plethora of fields, leading to dramatic reductions in the necessary number of measurements, samples, or computations. In our setting, the transition probabilities play the role of a target sparse signal of Fourier coefficients. The data reduction made possible via CS then translates to reducing necessary computations to a much smaller random subsample of PGF evaluations, which play the role of measurements used to recover the signal.

### 3.1 Overview

In the CS framework, the unknown signal is a vector  $\mathbf{x} \in \mathbb{C}^N$  observed through a measurement  $\mathbf{b} = \mathbf{V}\mathbf{x} \in \mathbb{C}^M$  with  $M \ll N$ . Here  $\mathbf{V}$  denotes an  $M \times N$  *measurement matrix* or sensing matrix. Since  $M < N$ , the system is underdetermined and inversion is highly ill-posed—the space of solutions is an infinite affine subspace, but CS theory shows that recovery can be accomplished under certain assumptions by seeking the *sparsest* solution. Let  $\psi$  be an orthonormal basis of  $\mathbb{C}^N$  that allows a  $K$ -sparse representation of  $\mathbf{x}$ : that is,  $\mathbf{x} = \psi\mathbf{s}$  where  $\mathbf{s}$  is a sparse vector of coefficients such that  $\|\mathbf{s}\|_0 < K$ . Candès [2006] proves that recovery can then be accurately accomplished by finding the sparsest solution

$$\hat{\mathbf{s}} = \underset{\mathbf{s}}{\operatorname{argmin}} \|\mathbf{s}\|_0 \quad \text{s.t.} \quad \mathbf{A}\mathbf{s} = \mathbf{b} \quad (7)$$

where  $\mathbf{A} = \mathbf{V}\psi$  is the composition of the measurement and sparsifying matrices. In practice, this non-convex objective is combinatorially intractable to solve exactly, and is instead solved by proxy via  $\ell_1$ -relaxation, resulting in a convex optimization program. In place of Equation (7), we optimize the unconstrained penalized objective

$$\hat{\mathbf{s}} = \underset{\mathbf{s}}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{A}\mathbf{s} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{s}\|_1, \quad (8)$$

where  $\lambda$  is a regularization parameter enforcing sparsity of  $\mathbf{s}$ . The signal  $\mathbf{x}$ , or equivalently  $\mathbf{s}$ , can be recovered perfectly using only  $M = CK \log N$  measurements for some constant  $C$  when  $\mathbf{A}$  satisfies the *Restricted Isometry Property* (RIP) [Candès and Tao, 2005, Candès, 2008]—briefly, this requires that  $\mathbf{V}$  and  $\psi$  to be *incoherent* so that rows of  $\mathbf{V}$  cannot sparsely represent the columns of  $\psi$  and vice versa. Coherence between  $\mathbf{V}$ ,  $\psi$  is defined as

$$\mu(\mathbf{V}, \psi) = \sqrt{n} \max_{i,j} |\langle \mathbf{V}_i, \psi_j \rangle|,$$

and low coherence pairs are desirable. It has been shown that choosing random measurements  $\mathbf{V}$  satisfies RIP with overwhelming probability [Candès, 2008]. Further, given

$\psi$ , it is often possible to choose a known ideal distribution from which to sample elements in  $\mathbf{V}$  such that  $\mathbf{V}$  and  $\psi$  are maximally incoherent.

### 3.2 Higher dimensions

CS theory extends naturally to higher-dimensional signals [Candès, 2006]. In the 2D case which will arise in our applications (Section 5), the sparse solution  $\mathbf{S} \in \mathbb{C}^{N \times N}$  and measurement

$$\mathbf{B} = \mathbf{A}\mathbf{S}\mathbf{A}^T \in \mathbb{C}^{M \times M} \quad (9)$$

are matrices rather than vectors, and we solve

$$\hat{\mathbf{S}} = \underset{\mathbf{S}}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{A}\mathbf{S}\mathbf{A}^T - \mathbf{B}\|_2^2 + \lambda \|\mathbf{S}\|_1. \quad (10)$$

This can always be equivalently represented in the vector-valued framework: vectorizing

$$\operatorname{vec}(\mathbf{S}) = \tilde{\mathbf{s}} \in \mathbb{C}^{N^2}, \quad \operatorname{vec}(\mathbf{B}) = \tilde{\mathbf{b}} \in \mathbb{C}^{M^2},$$

we now seek  $\tilde{\mathbf{b}} = \tilde{\mathbf{A}}\tilde{\mathbf{s}}$  as in Equations (7), (8), where  $\tilde{\mathbf{A}} = \mathbf{A} \otimes \mathbf{A}$  is the Kronecker product of  $\mathbf{A}$  with itself. In practice, it can be preferable to solve (10), since the number of entries in  $\tilde{\mathbf{A}}$  grows rapidly and thus the vectorized problem requires a costly construction of  $\tilde{\mathbf{A}}$  and can be cumbersome in terms of memory.

## 4 CSGF METHOD

We propose an algorithm that allows for efficient PGF inversion within a compressed sensing framework. We focus our exposition on two-type models: linear complexity in  $|\Omega|$  is less often a bottleneck in single-type problems, and all generating function methods as well as compressed sensing techniques we describe extend to settings with an arbitrary number of particle types.

We wish to compute the transition probabilities  $p_{jk,lm}(t)$  given any  $t > 0$  and  $\mathbf{X}(0) = (j, k)$ . These probabilities can be arranged in a matrix  $\mathbf{S} \in \mathbb{R}^{N \times N}$  with entries

$$\{\mathbf{S}\}_{l,m} = p_{jk,lm}(t).$$

Without the CS framework, these probabilities are obtained following Equation (6) by first computing an equally sized matrix of PGF solutions

$$\tilde{\mathbf{B}} = \left\{ \phi_{jk} \left( t, e^{\frac{2\pi i u}{N}}, e^{\frac{2\pi i v}{N}} \right) \right\}_{u,v=0}^{N-1} \in \mathbb{C}^{N \times N}. \quad (11)$$

For large  $N$ , obtaining  $\tilde{\mathbf{B}}$  is computationally expensive, and our method seeks to bypass this step. When  $\tilde{\mathbf{B}}$  is computed, transition probabilities are then recovered by taking the fast Fourier transform  $\mathbf{S} = \operatorname{fft}(\tilde{\mathbf{B}})$ . To better understand how this fits into the CS framework, we can equivalently write

the fast Fourier transform in terms of matrix operations  $\mathbf{S} = \mathbf{F}\tilde{\mathbf{B}}\mathbf{F}^T$ , where  $\mathbf{F} \in \mathbb{C}^{N \times N}$  denotes the discrete Fourier transform matrix (see Supplement). Thus, the sparsifying basis  $\psi$  is the Inverse Discrete Fourier Transform (IDFT) matrix  $\psi = \mathbf{F}^*$  given by the conjugate transpose of  $\mathbf{F}$ , and we have  $\tilde{\mathbf{B}} = \psi\mathbf{S}\psi^T$ .

When the solution matrix  $\mathbf{S}$  is expected to have a sparse representation, our CSGF method seeks to recover  $\mathbf{S}$  without computing the full matrix  $\tilde{\mathbf{B}}$ , instead beginning with a much smaller set of PGF evaluations  $\mathbf{B} \in \mathbb{C}^{M \times M}$  corresponding to random entries of  $\tilde{\mathbf{B}}$  selected uniformly at random. Denoting randomly sampled indices  $\mathcal{I}$ , this smaller matrix is a projection  $\mathbf{B} = \mathbf{A}\mathbf{S}\mathbf{A}^T$  in the form of Equation (9) where  $\mathbf{A} \in \mathbb{C}^{M \times N}$  is obtained by selecting a subset of rows of  $\psi$  corresponding to  $\mathcal{I}$ . Uniform sampling of rows corresponds to multiplying by a measurement matrix encoding the *spike basis* (or standard basis): formally, this fits into the framework described in Section 3.1 as  $\mathbf{A} = \mathbf{V}\psi$ , with measurement matrix rows  $\mathbf{V}_j(l) = \delta(j-l)$ . The spike and Fourier bases are known to be *maximally incoherent* in any dimension, so uniformly sampling indices  $\mathcal{I}$  is optimal in our setting.

Now in the compressed sensing framework, computing the reduced matrix  $\mathbf{B}$  only requires a logarithmic proportion  $|\mathbf{B}| \propto K \log |\tilde{\mathbf{B}}|$  of PGF evaluations necessary in Equation (11). Computing transition probabilities in  $\mathbf{S}$  is thus reduced to a signal recovery problem, solved by optimizing the objective in Equation (10).

#### 4.1 Solving the $\ell_1$ problem

There has been extensive research on algorithms for solving the  $\ell_1$  regularization objective in Equation (8) and related problems [Tibshirani, 1996, Beck and Teboulle, 2009a]. As mentioned previously, vectorizing the problem so that it can be represented in the form (8) requires wasteful extra memory; instead we choose to solve the objective in Equation (10) using a *proximal gradient descent* (PGD) algorithm.

PGD is useful for solving minimization problems with objective of the form  $f(x) = g(x) + h(x)$  with  $g$  convex and differentiable, and  $h$  convex but not necessarily differentiable. Letting

$$g(\mathbf{S}) = \frac{1}{2} \|\mathbf{A}\mathbf{S}\mathbf{A}^T - \mathbf{B}\|_2^2, \quad h(\mathbf{S}) = \lambda \|\mathbf{S}\|_1,$$

we see that Equation (10) satisfies these conditions. A form of generalized gradient descent, PGD iterates toward a solution with

$$x_{k+1} = \underset{z}{\operatorname{argmin}} [g(x_k) + \nabla g(x_k)^T(z - x_k) + \frac{1}{2L_k} \|z - x_k\|_2^2 + h(z)], \quad (12)$$

where  $L_k$  is a step size that is either fixed or determined via line-search. This minimization has known closed-form solution

$$x_{k+1} = \operatorname{softh}(x_k - L_k \nabla g(x_k), L_k \lambda), \quad (13)$$

where  $\operatorname{softh}$  is the soft-thresholding operator

$$[\operatorname{softh}(x, \alpha)]_i = \operatorname{sgn}(x_i) \max(|x_i| - \alpha, 0). \quad (14)$$

This results in an *iterative soft-thresholding algorithm* that solves the convex problem (10) with rate of convergence  $\mathcal{O}(1/k)$  when  $L_k$  is fixed. The  $\operatorname{softh}()$  operation is simple and computationally negligible, so that the main computational cost is in evaluating  $\nabla g(x_k)$ . We derive a closed form expression for the gradient in our setting:

$$\nabla g(\mathbf{S}) = -\mathbf{A}^*(\mathbf{B} - \mathbf{A}\mathbf{S}\mathbf{A}^T)\bar{\mathbf{A}}, \quad (15)$$

where  $\bar{\mathbf{A}}$ ,  $\mathbf{A}^*$  denote complex conjugate and conjugate transpose of  $\mathbf{A}$  respectively. In practice, the inner term  $\mathbf{A}\mathbf{S}\mathbf{A}^T$  is obtained as a subset of the inverse fast Fourier transform of  $\mathbf{S}$  rather than by explicit matrix multiplication. The computational effort in computing  $\nabla g(\mathbf{S})$  therefore involves only the two outer matrix multiplications.

We implement a fast variant of PGD using momentum terms [Beck and Teboulle, 2009b] based on an algorithm introduced by Nesterov, and select step sizes  $L_k$  via a simple line-search subroutine [Beck and Teboulle, 2009a]. The accelerated version includes an *extrapolation step*, where the soft-thresholding operator is applied to a momentum term

$$y_{k+1} = x_k + \omega_k(x_k - x_{k-1})$$

rather than to  $x_k$ ; here  $\omega_k$  is an extrapolation parameter for the momentum term. Remarkably, the accelerated method still only requires one gradient evaluation at each step as  $y_{k+1}$  is a simple linear combination of previously computed points, and has been proven to achieve the optimal worst-case rate of convergence  $\mathcal{O}(1/k^2)$  among first order methods [Nesterov, 1983]. Similarly, the line-search procedure involves evaluating a bound that also only requires one evaluation of  $\nabla g$  (see Supplement for further implementation details).

Algorithm 1 provides a summary of the CSGF method in pseudocode.

## 5 EXAMPLES

We will examine the performance of CSGF in two applications: a stochastic two-compartment model used in statistical studies of *hematopoiesis*, the process of blood cell production, and a birth-death-shift model that has been used to study the evolution of *transposons*, mobile genetic elements.

---

**Algorithm 1** CSGF algorithm.

- 1: **Input:** initial sizes  $X_1 = j, X_2 = k$ , time interval  $t$ , branching rates  $\theta$ , signal size  $N > j, k$ , measurement size  $M$ , penalization constant  $\lambda > 0$ , line-search parameters  $L, c$ .
  - 2: Uniformly sample  $M$  indices  $\mathcal{I} \subset [0, \dots, N - 1] / N$
  - 3: Compute  $\mathbf{B} = \{ \phi_{jk}(t, e^{2\pi i u / N}, e^{2\pi i v / N}) \}_{u, v \in \mathcal{I} \times \mathcal{I}}$
  - 4: Define  $\mathbf{A} = \psi_{\mathcal{I}}$  the  $\mathcal{I}$  rows of IDFT matrix  $\psi$
  - 5: **Initialize:**  $\mathbf{S}_1 = \mathbf{Y}_1 = \mathbf{0}$
  - 6: **for**  $k = 1, 2, \dots, \{\text{max iterations}\}$  **do**
  - 7:   Choose  $L_k = \text{line-search}(L, c, \mathbf{Y}_k)$
  - 8:   Update extrapolation parameter  $\omega_k = \frac{k}{k+3}$
  - 9:   Update momentum  $\mathbf{Y}_{k+1} = \mathbf{S}_k + \omega_k(\mathbf{S}_k - \mathbf{S}_{k-1})$
  - 10:   Compute  $\nabla g(\mathbf{Y}_{k+1})$  according to (15)
  - 11:   Update  $\mathbf{S}_{k+1} = \text{softh}(\mathbf{S}_k - L_k \nabla g(\mathbf{Y}_{k+1}), L_k \lambda)$
  - 12: **end for**
  - 13: **return**  $\hat{\mathbf{S}} = \mathbf{S}_{k+1}$
- 

### 5.1 Two-compartment hematopoiesis model

Hematopoiesis is the process in which self-sustaining primitive hematopoietic stem cells (HSCs) specialize, or *differentiate*, into progenitor cells, which further specialize to eventually produce mature blood cells. In addition to far-reaching clinical implications — stem cell transplantation is a mainstay of cancer therapy — understanding hematopoietic dynamics is biologically interesting, and provides critical insights of general relevance to other areas of stem cell biology [Orkin and Zon, 2008]. The stochastic model, depicted in Figure 1, has enabled estimation of hematopoietic rates in mammals from data in several studies [Catlin et al., 2001, Golinelli et al., 2006, Fong et al., 2009]. Without the ability to compute transition probabilities, an estimating equation approach by Catlin et al. [2001] is statistically inefficient, resulting in uncertain estimated parameters with very wide confidence intervals. Nonetheless, biologically sensible rates are inferred. Golinelli et al. [2006] observe that transition probabilities are unknown for a linear birth-death process (compartment 1) coupled with an inhomogeneous immigration-death process (compartment 2), motivating their computationally intensive reversible jump MCMC implementation.

However, we can equivalently view the model as a two-type branching process. Under such a representation, it becomes possible to compute transition probabilities via Equation (6). The type one particle population  $X_1$  corresponds to hematopoietic stem cells (HSCs), and  $X_2$  represents progenitor cells. With parameters as denoted in Figure 1, the nonzero instantaneous rates defining the process are

$$\begin{aligned} a_1(2, 0) &= \rho, & a_1(0, 1) &= \nu, & a_1(1, 0) &= -(\rho + \nu), \\ a_2(0, 0) &= \mu, & a_2(0, 1) &= -\mu. \end{aligned} \quad (16)$$

Having newly formulated the model as a two-type branch-

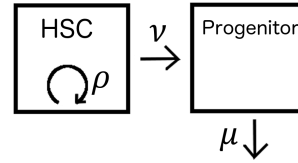


Figure 1: HSCs can self-renew, producing new HSCs at rate  $\rho$ , or differentiate into progenitor cells at rate  $\nu$ . Further progenitor differentiation is modeled by rate  $\mu$ .

ing process, we derive solutions for its PGF, defined in Equation (4), with details in the Supplement:

**Proposition 5.1** *The generating function for the two-type model described in (16) is given by  $\phi_{jk} = \phi_{1,0}^j \phi_{0,1}^k$ , where*

$$\begin{cases} \phi_{0,1}(t, s_1, s_2) = 1 + (s_2 - 1)e^{-\mu t} \\ \frac{d}{dt} \phi_{1,0}(t, s_1, s_2) = \rho \phi_{1,0}^2(t, s_1, s_2) - (\rho + \nu) \phi_{1,0}(t, s_1, s_2) \\ \quad + \nu \phi_{0,1}(t, s_1, s_2). \end{cases} \quad (17)$$

We see that  $\phi_{0,1}$  has closed form solution so that evaluating  $\phi_{jk}$  only requires solving one ODE numerically, and with the ability to compute  $\phi_{jk}$ , we may obtain transition probabilities using Equation (6). In this application, cell populations can easily reach thousands, motivating the CSGF approach to accelerate transition probability computations.

### 5.2 Birth-death-shift model for transposons

Our second application examines the birth-death-shift (BDS) process proposed by Rosenberg et al. [2003] to model evolutionary dynamics of transposable elements or *transposons*, genomic mobile sequence elements. Each transposon can (1) duplicate, with the new copy moving to a new genomic location; (2) shift to a different location; or (3) be removed and lost from the genome, independently of all other transposons. These respective birth, shift, and death events occur at per-particle instantaneous rates  $\beta, \sigma, \delta$ , with overall rates proportional to the total number of transposons. Transposons thus evolve according to a linear *birth-death-shift* Markov process in continuous time. In practice, genotyping technologies allow for this process to be discretely monitored, necessitating computation of finite-time transition probabilities.

Rosenberg et al. [2003] estimate evolutionary rates of the *IS6110* transposon in the *Mycobacterium tuberculosis* genome from a San Francisco community study dataset [Cattamanchi et al., 2006]. Without transition probabilities, the authors maximize an approximate likelihood by assuming at most one event occurs per observation interval, a rigid assumption that severely limits the range of applications. Doss et al. [2013] revisit their application, inferring similar rates of *IS6110* evolution using a one-dimensional birth-death model that ignores shift events. Xu et al. [2014] show that the BDS model over any finite observation interval can be modeled as a two-type branching process, where

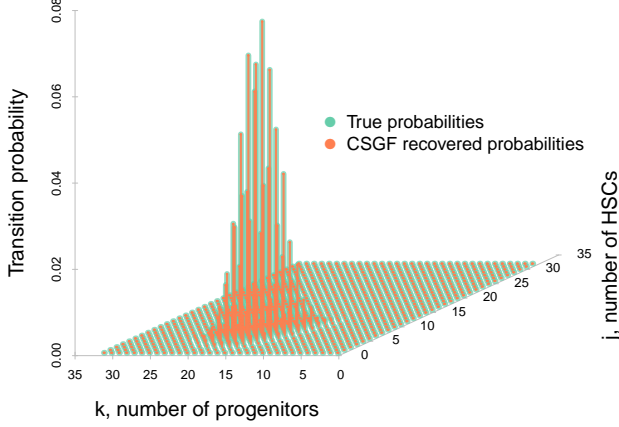


Figure 2: Illustrative example of recovered transition probabilities in hematopoiesis model described in Section 5. Beginning with 15 HSCs and 5 progenitors over a time period of one week, the CSGF solution  $\hat{\mathbf{S}} = \{\hat{p}_{(15,5),(j,k)}(1)\}$ ,  $j, k = 0, \dots, 31$ , perfectly recovers transition probabilities  $\mathbf{S}$ , using fewer than half the measurements.

$X_1$  denotes the number of initially occupied genomic locations and  $X_2$  denotes the number of newly occupied locations (see figure in Supplement). In this representation, full dynamics of the BDS model can be captured, and generating function techniques admit transition probabilities, leading to rate estimation via MLE and EM algorithms. Transposon counts in the tuberculosis dataset are low, so that Equation (6) can be computed easily, but their method does not scale well to applications with high counts in the data.

The nonzero rates defining the two-type branching process representation of the BDS model are given by

$$\begin{aligned} a_1(1, 1) &= \beta, & a_1(0, 1) &= \sigma, & a_1(0, 0) &= \delta, \\ a_1(1, 0) &= -(\beta + \sigma + \delta), & a_2(0, 2) &= \beta, \\ a_2(0, 1) &= -(\beta + \delta), & a_2(0, 0) &= \delta. \end{aligned} \quad (18)$$

and its PGF is governed by the following system derived in [Xu et al., 2014]:

$$\begin{cases} \phi_{0,1}(t, s_1, s_2) = 1 + \left[ \frac{\beta}{\delta - \beta} + \left( \frac{1}{s_2 - 1} + \frac{\beta}{\beta - \delta} \right) e^{(\delta - \beta)t} \right]^{-1} \\ \frac{d}{dt} \phi_{1,0}(t, s_1, s_2) = \beta \phi_{1,0} \phi_2 + \sigma \phi_{0,1} + \delta - (\beta + \sigma + \delta) s_1, \end{cases} \quad (19)$$

again with  $\phi_{jk} = \phi_{1,0}^j \phi_{0,1}^k$  by particle independence.

### 5.3 Results

To compare the performance of CSGF to the computation of Equation (6) without considering sparsity, we first compute sets of transition probabilities  $\mathbf{S}$  of the hematopoiesis model using the full set of PGF solution measurements  $\tilde{\mathbf{B}}$  as described in Equation (11). These “true signals” are

compared to the signals computed using CSGF  $\hat{\mathbf{S}}$ , recovered using only a random subset of measurements  $\mathbf{B}$  following Algorithm 1. Figure 2 provides an illustrative example with small cell populations for visual clarity—we see that the support of transition probabilities is concentrated (sparse), and the set of recovered probabilities  $\hat{\mathbf{S}}$  is visually identical to the true signal.

In each of the aforementioned applications, we calculate transition probabilities  $\mathbf{S} \in \mathbb{R}^{N \times N}$  for maximum populations  $N = 2^7, 2^8, \dots, 2^{12}$ , given rate parameters  $\theta$ , initial population  $\mathbf{X}(0)$ , and time intervals  $t$ . Each computation of  $\mathbf{S}$  requires  $N^2$  numerical evaluations of the ODE systems (17), (19). For each value of  $N$ , we repeat this procedure beginning with ten randomly chosen sets of initial populations  $\mathbf{X}(0)$  each with total size less than  $N$ . We compare the recovered signals  $\hat{\mathbf{S}}$  computed using CSGF to true signals  $\mathbf{S}$ , and report median runtimes and measures of accuracy over the ten trials, with details in the following sections.

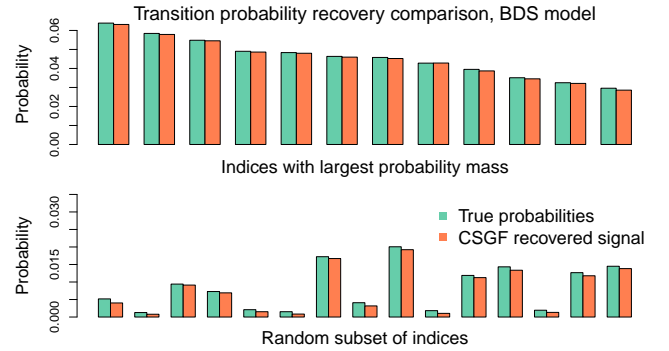


Figure 3: Randomly selected probabilities and largest probabilities recovered using CSGF are nearly identical to their true values. Probabilities displayed here correspond to a randomly selected BDS model trial with  $N=512$ ; transition probabilities  $\hat{\mathbf{S}}$  via CSGF are recovered from a sample  $\mathbf{B}$  requiring fewer than 2% of ODE computations used to compute  $\mathbf{S} = \text{fft}(\tilde{\mathbf{B}})$ .

**Parameter settings:** In the hematopoiesis example, we set per-week branching rates  $\theta_{\text{hema}} = (0.125, 0.104, 0.147)$  and observation time  $t = 1$  week based on biologically sensible rates and observation time scales of data from previous studies of hematopoiesis in mammals [Catlin et al., 2001, Golinelli et al., 2006, Fong et al., 2009]. For the BDS application, we set per-year event rates  $\theta_{\text{bds}} = (0.0156, 0.00426, 0.0187)$  estimated in [Xu et al., 2014], and  $t = 0.35$  years, the average length between observations in the San Francisco tuberculosis dataset [Cattamanchi et al., 2006].

In each case, we computed  $M^2 = 3K \log N^2$  total random measurements to obtain  $\mathbf{B}$  for CSGF, and we set the regu-



Table 1: Runtimes and error, hematopoiesis model. The third column reports total runtime of the generating function approach without using sparsity. Total runtimes using CSGF are the sum of the runtime for computing the subset of ODE solutions (fourth column) and runtime for PGD.

| $N$  | $M$ | Time (sec),<br>$\tilde{\mathbf{B}} \in \mathbb{C}^{N \times N}$ | Time (sec),<br>$\mathbf{B} \in \mathbb{C}^{M \times M}$ | Time (sec),<br>PGD | $\varepsilon_{\max} =$<br>$ \hat{p}_{ij,kl} - p_{ij,kl} _{\max}$ | $\varepsilon_{\text{rel}} =$<br>$\varepsilon_{\max} /  p_{ij,kl} _{\max}$ |
|------|-----|---|---|--------------------|--|---|
| 128  | 43  | 108.6   | 9.3   | 0.64               | $9.41 \times 10^{-4}$  | $2.25 \times 10^{-2}$   |
| 256  | 65  | 368.9   | 22.1  | 2.1                | $9.44 \times 10^{-4}$  | $4.73 \times 10^{-2}$   |
| 512  | 99  | 922.1   | 44.8  | 8.5                | $3.23 \times 10^{-4}$  | $3.60 \times 10^{-2}$   |
| 1024 | 147 | 5740.1  | 118.1   | 41.9               | $2.27 \times 10^{-4}$  | $5.01 \times 10^{-2}$   |
| 2048 | 217 | 12754.8   | 145.0   | 390.0              | $1.29 \times 10^{-4}$  | $5.10 \times 10^{-2}$   |
| 4096 | 322 | 58797.3   | 310.7   | 2920.3             | $9.43 \times 10^{-5}$  | $6.13 \times 10^{-2}$   |

Table 2: Runtimes and error, birth-death-shift model.

| $N$  | $M$ | Time (sec),<br>$\tilde{\mathbf{B}} \in \mathbb{C}^{N \times N}$ | Time (sec),<br>$\mathbf{B} \in \mathbb{C}^{M \times M}$ | Time (sec),<br>PGD | $\varepsilon_{\max} =$<br>$ \hat{p}_{ij,kl} - p_{ij,kl} _{\max}$ | $\varepsilon_{\text{rel}} =$<br>$\varepsilon_{\max} /  p_{ij,kl} _{\max}$ |
|------|-----|---|---|--------------------|--|---|
| 128  | 25  | 39.7  | 2.3   | 1.0                | $5.27 \times 10^{-3}$  | $2.77 \times 10^{-2}$   |
| 256  | 33  | 150.2   | 3.8   | 7.8                | $4.86 \times 10^{-3}$  | $4.71 \times 10^{-2}$   |
| 512  | 45  | 895.8   | 7.8   | 25.3               | $2.71 \times 10^{-3}$  | $4.68 \times 10^{-2}$   |
| 1024 | 68  | 2508.9  | 18.6  | 58.2               | $1.41 \times 10^{-3}$  | $5.12 \times 10^{-2}$   |
| 2048 | 101 | 9788.3  | 26.1  | 528.3              | $8.10 \times 10^{-4}$  | $4.81 \times 10^{-2}$   |
| 4096 | 150 | 40732.7   | 57.4  | 2234.7             | $4.01 \times 10^{-4}$  | $5.32 \times 10^{-2}$   |

larization parameters  $\lambda_{\text{hsc}} = \sqrt{\log M}$ ,  $\lambda_{\text{bds}} = \log M$ , with more regularization in the BDS application as lower rates and a shorter observation interval leads us to expect more sparsity. While careful case-by-case tuning to choose  $\lambda$ ,  $M$  would lead to optimal results, we set them in this simple manner across *all* trials to demonstrate a degree of robustness, still yielding promising performance results. In practice one may apply standard cross-validation procedures to select  $\lambda$ ,  $M$ , and because the target solution is a set of transition probabilities, checking that entries in the recovered solution  $\hat{\mathbf{S}}$  sum close to 1 offers a simpler available heuristic. Finally, though one may expedite convergence of PGD by supplying an informed initial guess with positive values near values  $\mathbf{X}(0)$  in practice, we initialize PGD with an uninformative initial value  $\mathbf{S}_1 = \mathbf{0}$  in all cases.

**Accuracy:** In both models and for all values of  $N$ , each signal was reconstructed very accurately. Errors are reported in Tables 1 and 2 for the hematopoiesis and BDS models respectively. Maximum absolute errors for each CSGF recovery

$$\varepsilon_{\max} = \max_{kl} |\{\hat{\mathbf{S}}\}_{kl} - \{\mathbf{S}\}_{kl}| = \max_{kl} |\hat{p}_{ij,kl}(t) - p_{ij,kl}(t)|$$

are on the order of  $10^{-3}$  at worst. Because  $\varepsilon_{\max}$  is typically attained at large probabilities, we include the maximum absolute error relative to the largest transition probability

$$\varepsilon_{\text{rel}} = \frac{\varepsilon_{\max}}{\max_{kl} \{S\}_{kl}},$$

providing a more conservative measure of accuracy. We still see that  $\varepsilon_{\text{rel}}$  is on the order of  $10^{-2}$  in all cases. Addi-

tional analysis and discussion of accuracy in terms of relative error are included in the Supplement, although the measures here arguably give more insight to the performance of our algorithm. Visually, the accuracy of CSGF is stark: Figure 3 provides a side-by-side comparison of randomly selected transition probabilities recovered in the BDS model for  $N = 2^9$ .

**Running Times:** Tables 1 and 2 show dramatic improvements in runtime using CSGF, reducing the number of ODE computations logarithmically. For instance, with  $N = 4096$ , we see the time spent on PGF evaluations necessary for CSGF is less than 0.1% of the time required to compute  $\mathbf{S}$  in the BDS model, and around 0.5% of computational cost in the less sparse hematopoiesis application. Including the time required for solving Equation (10) via PGD, we see that computing  $\hat{\mathbf{S}}$  using CSGF reduces runtime by two orders of magnitude, requiring less than 6% of total computational time spent toward computing  $\mathbf{S}$  in the worst case. We remark that ODE solutions are computed using a C implementation of efficient solvers via package `deSolve`, while we employ a naive R implementation of PGD. We emphasize the logarithmic reduction in required numerical ODE solutions; an optimized implementation of PGD reducing R overhead will yield further real-time efficiency gains.

## 6 DISCUSSION

We have presented a novel adaptation of recent generating function techniques to compute branching process transi-

tion probabilities within the compressed sensing paradigm. While generating function approaches bypass costly matrix exponentiation and simulation-based techniques by exploiting mathematical properties in the branching structure, our contribution now makes these techniques scalable by additionally harnessing the available sparsity structure. We show that when sparsity is present in the set of transition probabilities, computational cost can be reduced up to a logarithmic factor over existing methods. Note that sparsity is the *only* additional assumption necessary to apply our CSGF method—no prior knowledge about where transition probabilities have support is necessary. Further, while our algorithm uses proximal gradient descent to solve the resulting constrained optimization problem, the framework we propose is very general, and users may choose among many standard optimization techniques to best suit their application.

Many real-world applications of branching process modeling feature such sparsity, and we have seen that CSGF achieves accurate results with significant efficiency gains in two such examples with realistic parameter settings from the scientific literature. Transition probabilities are often important, interpretable quantities in their own right, and are necessary within any likelihood-based probabilistic framework for partially observed CTMCs. Their tractability using CSGF opens doors to applying Bayesian and frequentist tools alike to settings in which such methods were previously infeasible. Finally, we note that other statistically relevant quantities such as expectations of particle dwell times and restricted moments can be computed using similar generating function techniques [Minin and Suchard, 2008], and the CSGF framework applies analogously when sparsity is present.

**Acknowledgements** We thank Alan Mackey, Hari Narayanan, and Noah Simon for helpful discussions and guidance. VNM was supported by NIH grants R01-AI107034 and U54-GM111274. JX was supported by an NDSEG fellowship.

## References

- C Andrieu, A Doucet, and R Holenstein. Particle Markov chain Monte Carlo methods. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 72(3):269–342, 2010.
- NTJ Bailey. The Elements of Stochastic Processes; with Applications to the Natural Sciences. New York: Wiley, 1964.
- A Beck and M Teboulle. Gradient-based algorithms with applications to signal recovery. In Palomar DP and Eldar YC, editors, Convex Optimization in Signal Processing and Communications. Cambridge University Press, Cambridge, UK, 2009a.
- A Beck and M Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM Journal on Imaging Sciences, 2(1):183–202, 2009b.
- JC Butcher. The Numerical Analysis of Ordinary Differential Equations: Runge-Kutta and General Linear Methods. Wiley-Interscience, 1987.
- EJ Candès. Compressive sampling. In Proceedings of the International Congress of Mathematicians: Madrid, August 22-30, 2006: invited lectures, pages 1433–1452, 2006.
- EJ Candès. The restricted isometry property and its implications for compressed sensing. Comptes Rendus Mathématique, 346(9):589–592, 2008.
- EJ Candès and T Tao. Decoding by linear programming. IEEE Transactions on Information Theory, 51(12):4203–4215, 2005.
- SN Catlin, JL Abkowitz, and P Guttorp. Statistical inference in a two-compartment model for hematopoiesis. Biometrics, 57(2):546–553, 2001.
- A Cattamanchi, PC Hopewell, LC Gonzalez, DH Osmond, L Masae Kawamura, CL Daley, and RM Jasmer. A 13-year molecular epidemiological analysis of tuberculosis in San Francisco. The International Journal of Tuberculosis and Lung Disease, 10(3):297–304, 2006.
- FW Crawford, VN Minin, and MA Suchard. Estimation for general birth-death processes. Journal of the American Statistical Association, 109(506):730–747, 2014.
- DL Donoho. Compressed sensing. IEEE Transactions on Information Theory, 52(4):1289–1306, 2006.
- CR Doss, Ma Suchard, I Holmes, MM Kato-Maeda, and VN Minin. Fitting birth–death processes to panel data with applications to bacterial DNA fingerprinting. The Annals of Applied Statistics, 7(4):2315–2335, 2013.
- A Doucet, S Godsill, and C Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. Statistics and computing, 10(3):197–208, 2000.
- T El-Hay, N Friedman, D Koller, and R Kupferman. Continuous time Markov networks. In Proceedings of the Twenty-second Conference on Uncertainty in AI (UAI), Boston, Massachusetts, July 2006.
- Y Fong, P Guttorp, and J Abkowitz. Bayesian inference and model choice in a hidden stochastic two-compartment model of hematopoietic stem cell fate decisions. The Annals of Applied Statistics, 3(4):1695–1709, 12 2009.
- D Golinelli, P Guttorp, and JA Abkowitz. Bayesian inference in a hidden stochastic two-compartment model for feline hematopoiesis. Mathematical Medicine and Biology, 23(3):153–172, 2006.

- WK Grassmann. Transient solutions in Markovian queueing systems. Computers & Operations Research, 4(1): 47–53, 1977.
- P Guttorp. Stochastic modeling of scientific data. CRC Press, 1995.
- M Hajiaghayi, B Kirkpatrick, L Wang, and A Bouchard-Côté. Efficient continuous-time Markov chain estimation. In Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014, pages 638–646, 2014.
- K Lange. Calculation of the equilibrium distribution for a deleterious gene by the finite Fourier transform. Biometrics, 38(1):79–86, 1982.
- VN Minin and MA Suchard. Counting labeled transitions in continuous-time Markov models of evolution. Journal of Mathematical Biology, 56(3):391–412, 2008.
- Y Nesterov. A method of solving a convex programming problem with convergence rate  $O(1/k^2)$ . Soviet Mathematics Doklady, 27(2):372–376, 1983.
- SH Orkin and LI Zon. Hematopoiesis: An evolving paradigm for stem cell biology. Cell, 132(4):631–644, 2008.
- VA Rao and YW Teh. Fast MCMC sampling for Markov jump processes and continuous time Bayesian networks. In Proceedings of the 27th International Conference on Uncertainty in Artificial Intelligence. 2011.
- E Renshaw. Stochastic Population Processes: Analysis, Approximations, Simulations. Oxford University Press Oxford, UK, 2011.
- NA Rosenberg, AG Tsolaki, and MM Tanaka. Estimating change rates of genetic markers using serial samples: applications to the transposon *IS6110* in *Mycobacterium tuberculosis*. Theoretical Population Biology, 63(4):347–363, 2003.
- SM Ross. Approximating transition probabilities and mean occupation times in continuous-time Markov chains. Probability in the Engineering and Informational Sciences, 1(03):251–264, 1987.
- CE Shannon. A mathematical theory of communication. ACM SIGMOBILE Mobile Computing and Communications Review, 5(1):3–55, 2001.
- R Tibshirani. Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society. Series B (Methodological), pages 267–288, 1996.
- J Xu, P Guttorp, MM Kato-Maeda, and VN Minin. Likelihood-based inference for discretely observed birth-death-shift processes, with applications to evolution of mobile genetic elements. ArXiv e-prints, arXiv:1411.0031, 2014.

---

# Extend Transferable Belief Models with Probabilistic Priors

---

**Chunlai Zhou**

Computer Science Department  
School of Information  
Renmin University of China  
Beijing, CHINA 100872  
czhou@ruc.edu.cn

**Yuan Feng**

Centre for Quantum Computation and Intelligent Systems  
Faculty of Engineering and Information technology  
University of Technology, Sydney  
Broadway, NSW 2007 AUSTRALIA  
Yuan.Feng@uts.edu.au

## Abstract

In this paper, we extend Smets' *transferable belief model* (TBM) with *probabilistic* priors. Our first motivation for the extension is about evidential reasoning when the underlying prior knowledge base is Bayesian. We extend standard Dempster models with prior probabilities to represent beliefs and distinguish between two types of induced mass functions on an extended Dempster model: one for believing and the other essentially for decision-making. There is a natural correspondence between these two mass functions. In the extended model, we propose two conditioning rules for evidential reasoning with probabilistic knowledge base. Our second motivation is about the partial dissociation of betting at the pignistic level from believing at the credal level in TBM. In our *extended* TBM, we coordinate these two levels by employing the extended Dempster model to represent beliefs at the credal level. Pignistic probabilities are derived not from the induced mass function for believing but from the one for decision-making in the model and hence need not rely on the choice of frame of discernment. Moreover, we show that the above two proposed conditionings and marginalization (or coarsening) are consistent with pignistic transformation in the extended TBM.

## 1 INTRODUCTION

Reasoning about uncertainty is a fundamental issue for Artificial Intelligence [HALPERN, 2005]. Numerous approaches have been proposed, including the Dempster-Shafer theory of belief functions [SHAFER, 1976] (also called the theory of evidence or simply DS theory). Ever since the pioneering works by Dempster and Shafer, the theory of belief functions has become a powerful formalism in Artificial Intelligence for knowledge representation

and decision-making.

The *transferable belief model* (TBM) is a model developed to justify the use of belief functions (including Dempster's rule of combination) to model someone's beliefs [SMETS AND KENNES, 1994]. A TBM  $M = \langle (\Omega, m), Betp \rangle$  is a two-level mental model which distinguishes between two aspects of beliefs on a frame of discernment  $\Omega$ , beliefs for weighted opinions, and beliefs for decision making. The two levels are: the credal level, where beliefs are entertained and represented by a mass function  $m$ , and the pignistic level, where beliefs are used to make decisions and quantified as a probability distribution  $Betp_m$ , which is derived from mass function  $m$  by the so-called *pignistic transformation* (usually denoted by  $Betp$ ). The justification for the use of pignistic probabilities is usually linked to "rational" behavior exhibited by an ideal agent involved in some betting or decision contexts. But those probabilities do not represent the agent's beliefs; they are only the functions needed to derive the best decision. Let's consider a motivating example in TBM [SMETS AND KENNES, 1994].

**Example 1.1** (Betting under total ignorance) Consider a guard in a huge power plant. On the emergency panel, alarms  $A_1$  and  $A_2$  are both on. The guard never heard about these two alarms. He takes the instruction book and discovers that  $A_1$  is on iff circuit  $C$  is in state  $C_1$  or  $C_2$  and that alarm  $A_2$  is on iff circuit  $D$  is in state  $D_1, D_2$  or  $D_3$ . He never heard about these  $C$  and  $D$  circuits. There, his beliefs on the  $C$  circuit will be characterized by a "vacuous" belief function  $bel^{\Omega_C}$  on space  $\Omega_C = \{C_1, C_2\}$ , i.e., a belief function whose mass function satisfies  $m^{\Omega_C}(\Omega_C) = 1$  (this particular belief function is the one that represents the state of total ignorance). By the application of pignistic transformation, his pignistic probabilities will be given by

$$Betp^{\Omega_C}(C_1) = Betp^{\Omega_C}(C_2) = \frac{1}{2}.$$

Similarly, for the  $D$  circuit, the guard's belief  $bel^{\Omega_D}$  on the space  $\Omega_D = \{D_1, D_2, D_3\}$  will be vacuous, i.e., its corresponding mass function  $m^{\Omega_D}(\Omega_D) = 1$ , and the pignistic probabilities are

$$\text{Betp}^{\Omega_D}(D_1) = \text{Betp}^{\Omega_D}(D_2) = \text{Betp}^{\Omega_D}(D_3) = \frac{1}{3}.$$

Now by reading the next page on the manual, the guard discovers that circuits  $C$  and  $D$  are so made that whenever  $C$  is in the state  $C_1$ , circuit  $D$  is in state  $D_1$  and vice versa. So he learns that  $C_1$  and  $D_1$  are equivalent (given what the guard knows) and that  $C_2$  and  $(D_2 \text{ or } D_3)$  are equivalent. In the TBM, this information does not modify his belief about which circuit is broken. Within the transferable belief model, the only requirement is that equivalent propositions should receive equal beliefs (it is satisfied as  $\text{bel}^{\Omega_C}(C_1) = \text{bel}^{\Omega_D}(D_1) = 0$ ). Pignistic probabilities depend not only on these beliefs but also on *the structure of the betting frame*. In contrast, according to Bayesian approach, equivalent propositions should receive identical beliefs and therefore identical probabilities. However,  $\text{Betp}^{\Omega_C}(C_1) = \frac{1}{2}$  and  $\text{Betp}^{\Omega_D}(D_1) = \frac{1}{3}$  although  $\text{bel}^{\Omega_C}(C_1) = \text{bel}^{\Omega_D}(D_1) = 0$ .

The fact that the TBM can cope easily with such states of ignorance results from the partial dissociation between the credal and the pignistic levels. But this kind of separation of betting from believing makes the TBM vulnerable to Dutch books in decision-making [SNOW, 1998].

In this paper, we extend Smets' TBM with a probabilistic prior to coordinate reasoning at the credal and pignistic levels. Our first motivation is about evidential reasoning when the underlying prior knowledge base is Bayesian. In order to incorporate the influence of the Bayesian knowledge base, we extend standard Dempster models, which are used for representing belief functions, with probabilistic priors. For an extended Dempster model  $M$  with a prior probability  $pr$ , there are two induced mass functions. The first one  $m_D$  is derived in the standard way from the Dempster part  $D$  of  $M$  without the prior probability and hence complies with the well-known DS theory, especially with Dempster's rule of combination. The second  $m_M$  is induced by combining  $m_D$  with the prior probability  $pr$ . Conversely,  $m_D$  can be obtained from  $m_M$  by removing the influence of  $pr$ . So, there is a natural correspondence between  $m_D$  and  $m_M$ . However, these two mass functions are essentially different:  $m_D$  measures the belief update and  $m_M$  absolute belief or weighted opinion. We propose a new combination rule for the mass functions  $m_M$ 's which incorporate prior probabilities. The new combination rule is shown to be parallel to Dempster's rule for the mass functions  $m_D$ 's without the influence of prior probabilities. According to the new combination rule, we provide two *prediction-style* conditioning rules: one for *certain* conditioning knowledge and the other for *uncertain* knowledge.

Our second motivation is to coordinate reasoning at the credal and pignistic levels. We extend Smets' TBM by employing an extended Dempster model  $M$  to represent beliefs at the credal level and provide a corresponding generalized pignistic transformation  $\text{Betp}$  for this extended TBM. We

prove that the above two new conditioning rules in  $M$  are consistent with this pignistic transformation. In our extended TBM, since beliefs are represented by the induced mass function  $m_D$  of the Dempster part of  $M$ , they are insensitive to the choice of frame. Pignistic probabilities are derived not from the induced mass function  $m_D$  of the Dempster part of  $M$  but from the induced mass function  $m_M$ , which have incorporated the prior probability  $pr$ . We show by transforming the prior probability that pignistic probabilities obtained in this way need not rely on the choice of frame of discernment.

## 2 BASIC DEFINITIONS AND NOTIONS

Let  $\Omega$  be a frame of discernment and  $\mathcal{A} = 2^\Omega$  be the Boolean algebra of events. A *mass function* (or *mass assignment*) is a mapping  $m : \mathcal{A} \rightarrow [0, 1]$  satisfying  $\sum_{A \in \mathcal{A}} m(A) = 1$ . A mass function  $m$  is called *normal* if  $m(\emptyset) = 0$ . Without further notice, all mass functions in this paper are assumed to be normal. A set is called *focal* if  $m(A) > 0$ . A mass function  $m$  is called *categorical* if it has only one focal set. A *belief function* is a function  $\text{bel} : \mathcal{A} \rightarrow [0, 1]$  satisfying the following conditions:

1.  $\text{bel}(\emptyset) = 0, \text{bel}(\Omega) = 1$ ; and
2.  $\text{bel}(\bigcup_{i=1}^n A_i) \geq \sum_{\emptyset \neq I \subseteq \{1, \dots, n\}} (-1)^{|I|+1} \text{bel}(\bigcap_{i \in I} A_i)$   
where  $A_i \in \mathcal{A}$  for all  $i \in \{1, \dots, n\}$ .

A mapping  $f : \mathcal{A} \rightarrow [0, 1]$  is a belief function if and only if its Möbius transform is a mass function [SHAFER, 1976]. In other words, if  $m : \mathcal{A} \rightarrow [0, 1]$  is a mass function, then it determines a belief function  $\text{bel} : \mathcal{A} \rightarrow [0, 1]$  as follows:  $\text{bel}(A) = \sum_{B \subseteq A} m(B)$  for all  $A \in \mathcal{A}$ . Moreover, given a belief function  $\text{bel}$ , we can obtain its corresponding mass function  $m$  as follows:  $m(A) = \sum_{B \subseteq A} (-1)^{|A \setminus B|} \text{bel}(B)$  for all  $A \in \mathcal{A}$ . Intuitively, for a subset event  $A$ ,  $m(A)$  measures the belief that an agent commits *exactly* to  $A$ , not the total belief  $\text{bel}(A)$  that an agent commits to  $A$ . The corresponding *plausibility function*  $pl : 2^\Omega \rightarrow [0, 1]$  is dual to  $\text{bel}$  in the sense that  $pl(A) = 1 - \text{bel}(\bar{A})$  for all  $A \subseteq \Omega$ . If  $m_1$  and  $m_2$  are two mass functions on  $\Omega$  induced by two *independent* evidential sources, the combined mass function is calculated according to *Dempster's rule of combination*: for any  $C \subseteq \Omega$ ,

$$(m_1 \oplus m_2)(C) = \frac{\sum_{A \cap B = C} m_1(A) m_2(B)}{\sum_{A \cap B \neq \emptyset} m_1(A) m_2(B)} \quad (1)$$

When an event  $E$  is observed, then the conditional mass function of  $m$  is obtained according to *Dempster conditioning*: for any  $C \subseteq \Omega$ ,

$$m(C | E) = \frac{\sum_{B \cap E = C} m(B)}{pl(E)} \quad (2)$$

A transferable belief model  $M = \langle (\Omega, m), \text{Betp} \rangle$  [SMETS AND KENNES, 1994] is a two-level mental model: the credal level where beliefs are represented by a mass function  $m$ , and the pignistic level where decisions are made by maximizing expected utility. Hence we must build a probability distribution to compute these expectations. This probability distribution is *based on* the agent's beliefs, but should not be understood as representing the agent's beliefs. It is just a probability distribution *derived from* the mass function through *pignistic transformation*  $\text{Betp}$ . The pignistic transformation for the above mass function  $m$  is given by

$$\text{Betp}_m(\{\omega\}) := \sum_{\omega \in B \subseteq \Omega} \frac{1}{|B|} m(B) \text{ for any } \omega \in \Omega.$$

Note that  $\text{Betp}_m$  is a probability distribution on  $\Omega$  and is called a *pignistic probability distribution*. When the context is clear, we usually use  $m$  to denote the belief model  $M$ .

In order to show the sensitivity of pignistic transformation to the choice of frames of discernment, we need to set up a setting in terms of *refinements and coarsenings* of frames of discernment. The idea that one frame  $\Omega$  of discernment is obtained from another frame  $\Theta$  by splitting some or all of the elements of  $\Theta$  may be represented mathematically by specifying, for each  $\theta \in \Theta$ , the subset  $\omega(\{\theta\})$  of  $\Omega$  consisting of those possibilities into which  $\theta$  has been split. Such a mapping  $\omega$  is called a *refining*. Whenever  $\omega : 2^\Theta \rightarrow 2^\Omega$  is a refining, we call  $\Omega$  a *refinement* of  $\Theta$  and  $\Theta$  a *coarsening* of  $\Omega$ . In this paper, we are particularly interested in the case when  $\Theta$  is the set of equivalence classes with respect to some partition  $\Pi$  of  $\Omega$ . So the mapping  $\omega(\{\Pi(w)\}) = \Pi(w)$  for each  $w \in \Omega$  is a refinement and  $\Theta$  is a coarsening of  $\Omega$  where  $\Pi(w)$  is the equivalence class of  $w$ . We denote this special coarsening  $\Theta$  of  $\Omega$  as  $\Omega/\Pi$ . On the other hand,  $\Omega/\Pi$  may be regarded as a subalgebra  $\mathcal{B}$  of the powerset of  $\Omega$  with the set of *atoms* forming the partition  $\Pi$  of  $\Omega$ . In the following sections, we won't distinguish between  $\Omega/\Pi$  and  $\langle \Omega, \mathcal{B} \rangle$ . For each  $A \subseteq \Omega$ , we define  $\mathbf{B}(A) := \bigcap \{B \in \mathcal{B} : A \subseteq B\}$ . In other words,  $\mathbf{B}(A)$  is the least element of  $\mathcal{B}$  that contains  $A$  as a subset and hence is called the *upper approximation* of  $A$  in  $\mathcal{B}$ . For example,  $\Pi = \{\{w_1\}, \{w_2, w_3\}, \{w_4, w_5, w_6\}\}$  is a partition of  $\Omega = \{w_1, w_2, w_3, w_4, w_5, w_6\}$ . Then the associated subalgebra  $\mathcal{B}$  consists of the sets  $\bigcup_{B \subseteq \Pi} B$  with the atoms  $\{w_1\}$ ,  $\{w_2, w_3\}$ , and  $\{w_4, w_5, w_6\}$  in  $\mathcal{B}$ . If  $A = \{w_1, w_3, w_5\}$ , then  $\mathbf{B}(A) = \Omega$ .

Let  $\langle \Omega, \mathcal{B} \rangle$  be a coarsening of  $\Omega$  where  $\mathcal{B}$  is a subalgebra of the powerset  $2^\Omega$  with its atoms forming a partition of  $\Omega$ . Each element  $B$  of  $\mathcal{B}$  is a disjoint union of some atoms in  $\mathcal{B}$ . Suppose that  $bel : 2^\Omega \rightarrow [0, 1]$  is a belief function on  $\Omega$  with  $m$  as its corresponding mass function. Then *the derived mass function*  $m_{\mathcal{B}}$  on the coarsening  $\langle \Omega, \mathcal{B} \rangle$  can be obtained through the formula: for any  $B \in \mathcal{B}$ ,  $m_{\mathcal{B}}(B) = \sum_{\mathbf{B}(A)=B, A \subseteq \Omega} m(A)$ . Let  $bel_{\mathcal{B}}$  denote

the corresponding belief function. It is easy to check that, for any  $B \in \mathcal{B}$ ,  $bel_{\mathcal{B}}(B) = bel(B)$ . Intuitively,  $bel_{\mathcal{B}}$  is the derived belief function on the coarsening frame of discernment with less distinctions. The beliefs in the same propositions in these two frames with different distinctions should be equal. In this sense, believing in terms of belief functions is insensitive to the choice of frame of discernment.

### 3 EXTENDED DEMPSTER MODELS

In order to motivate our work of extending Smets' transferable belief models with probabilistic priors, we first represent belief functions through Dempster models.

#### 3.1 EXTENDED DEMPSTER MODELS

**Definition 3.1** A *Dempster model* is a tuple  $\langle (U, Pr), \Gamma, \Omega \rangle$  where  $(U, Pr)$  is a probability space and  $\Gamma$  is a multivalued mapping from  $U$  to  $\Omega$ , i.e., a mapping from  $U$  to  $2^\Omega$ , the powerset of  $\Omega$ .  $\triangleleft$

The multivalued mapping  $\Gamma$  is essentially a random subset on  $\Omega$ , and it induces a mass function  $m$  on  $\Omega$ :  $m(A) := Pr(\Gamma^{-1}(A))$  for any  $A \subseteq \Omega$ . We have the corresponding belief function  $Bel(A) = \sum_{B \subseteq A} Pr(\Gamma^{-1}(B))$ . Conversely, any mass function on  $\Omega$  can be represented as the induced mass function of some Dempster model. Before we extend Dempster models with probabilistic priors on  $\Omega$ , we use the well-known three prisoner paradox to show the necessity of the probabilistic priors.

**Example 3.2** (The Three Prisoners Paradox [HALPERN, 2005]) Of three prisoners  $a, b$  and  $c$ , only one of them is to be executed but  $a$  does not know which one. He therefore says to the jailer, "Since either  $b$  and  $c$  is certainly going to be declared innocent, you will give me no information about my chances if you give me the name of one man, either  $b$  or  $c$ , who is going to be freed." Accepting this argument, the jailer truthfully replies, " $b$  will be freed." Thereupon  $a$  feels sad because of the Bayesian conditioning on  $U := \{a, b, c\}$ : before the jailer replied, his own chances of being executed was one-third, but afterwards there are only two people, himself and  $c$ , who could be the one being executed, and so his chances of execution increases and is one-half.

Is  $a$  justified in believing that his chances of being executed have increased? Now we formulate this problem in the framework of a Dempster model. Consider the set of all possible outcomes:  $\Omega := \{(a, b), (a, c), (b, c), (c, b)\}$  where, for example,  $(a, b)$  means that  $a$  is to be executed and the jailer says that  $b$  will be freed. Suppose that at first  $a$  assumes that the initial decision as to who will be executed is made at random but assumes *nothing* about how the jailer will act except that he will tel-

l the truth. Let the random choice of who will be executed be represented by the probability space  $(U, Pr)$  where  $Pr$  is the uniform distribution on  $U$ . A multivalued mapping  $\Gamma : U \rightarrow 2^\Omega$  for delineating the possible outcomes when  $a, b$  or  $c$  is to be executed is given by:  $\Gamma(a) = \{(a, b), (a, c)\}, \Gamma(b) = \{(b, c)\}, \Gamma(c) = \{(c, b)\}$ . So the induced mass function  $m$  at the credal level is given by:  $m(\{(a, b), (a, c)\}) = m(\{(b, c)\}) = m(\{(c, b)\}) = \frac{1}{3}$ . Let  $E_a$  denote the event that  $a$  will be executed and  $J_b$  the event that the jailer says that  $b$  will be freed. Then  $E_a = \{(a, b), (a, c)\}$  and  $J_b = \{(a, b), (c, b)\}$ . According to Dempster's rule of conditionalization, we get that  $Bel(E_a | J_b) = Pl(E_a | J_b) = \frac{1}{2}$ . So Dempster's conditioning provides the same answer as that by the above  $a$ 's conditioning on the "naive" space  $U$  according to Bayesian rule [GRÜNWARD AND HALPERN, 2003]. By applying Smets' pignistic transformation, we obtain its probability distribution at the pignistic level:  $Betp_m(a, b) = Betp_m(a, c) = 1/6$  and  $Betp_m(b, c) = Betp_m(c, b) = 1/3$ .

More generally, we may assume that the jailer will tell the truth and  $a$ 's knowledge about the jailer's preference over his possible choices is formulated by a probabilistic prior on  $\Omega$ , which is *independent* of the assumption that the executed prisoner is chosen at random. Now we extend standard Dempster models by incorporating this kind of probabilities and express the induced beliefs at the credal level.

**Definition 3.3** An *extended Dempster-model*  $M = \langle (U, Pr), \Gamma, (\Omega, pr) \rangle$  is a Dempster model  $\langle (U, Pr), \Gamma, \Omega \rangle$  plus a prior probability  $pr$  on  $\Omega$  where  $pr$  is independent of  $\Gamma$  with respect to  $Pr$ .  $\triangleleft$

Now we explain this independence through a representation result of extended Dempster models.

**Lemma 3.4** Every *extended Dempster model*  $M = \langle (U, Pr), \Gamma, (\Omega, pr) \rangle$  can be represented as a *standard Dempster model*  $\langle (U', Pr'), \Gamma', \Omega \rangle$  with an additional mapping  $\gamma'$  from  $U'$  to  $\Omega$  for some probability space  $(U', Pr')$  and some multivalued mapping  $\Gamma'$  from  $U'$  to  $\Omega$ .

**Proof.** For a given extended Dempster model  $M = \langle (U, Pr), \Gamma, (\Omega, pr) \rangle$ , we define a new probability space  $(U', Pr')$ , which is essentially the Cartesian product of  $(U, Pr)$  and  $(\Omega, pr)$ , as follows:

- $U' = U \times \Omega$ ;
- $Pr'(x, y) = Pr(x)pr(y)$  for any  $(x, y) \in U'$ .

Further we define a multivalued mapping  $\Gamma' : U' \rightarrow 2^\Omega$  and a mapping  $\gamma' : U' \rightarrow \Omega$  as follows:

- $\Gamma'(x, y) = \Gamma(x)$ ,
- $\gamma'(x, y) = y$  for any  $(x, y) \in U'$ .

It is easy to check that  $Pr'((\Gamma')^{-1}(A)) = Pr(\Gamma^{-1}(A))$ , and  $Pr'((\gamma')^{-1}(A)) = pr(A)$  for any  $A \subseteq \Omega$ .  $\square$

So, in the following sections of this paper, we won't distinguish these two forms of extended Dempster models and will sometimes write an extended Dempster model as  $M = \langle (U, Pr), \Gamma, \gamma, \Omega \rangle$  where  $\langle (U, Pr), \Gamma, \Omega \rangle$  is a standard Dempster model and  $\gamma$  is a mapping from  $U$  to  $\Omega$ . In  $M$ , the prior probability  $pr$  is obtained by  $pr(A) = Pr(\{u \in U : \gamma(u) \in A\})$ . In this paper,  $\Gamma = A$  is shorthand for the event  $\{u \in U : \Gamma(u) = A\}$ ,  $\gamma \in A$  for  $\{u \in U : \gamma(u) \in A\}$  and  $\gamma \in \Gamma$  denotes  $\{u \in U : \gamma(u) \in \Gamma(u)\}$ . In  $M$ , the independence of the prior probability  $pr$  of the multivalued mapping  $\Gamma$  with respect to  $Pr$  means the independence of  $\gamma$  and  $\Gamma$ : for any subsets  $A$  and  $B$  of  $\Omega$ ,

$$Pr(\Gamma = A \wedge \gamma \in B) = Pr(\Gamma = A)Pr(\gamma \in B).$$

Just as in a Dempster model, we associate each extended Dempster model  $M = \langle (U, Pr), \Gamma, \gamma, \Omega \rangle$  with a mapping  $m_M : 2^\Omega \rightarrow [0, 1]$  which incorporates the mapping  $\gamma$  as follows:

$$m_M(A) := Pr(\Gamma = A | \gamma \in \Gamma) \quad (3)$$

It is easy to see that, since  $\Gamma$  and  $\gamma$  are independent with respect to  $Pr$ ,  $Pr(\gamma \in \Gamma) = \sum_{A \subseteq \Omega} Pr(\gamma \in A \wedge \Gamma = A) = \sum_{A \subseteq \Omega} Pr(\gamma \in A)Pr(\Gamma = A)$ . And  $Pr(\gamma \in \Gamma)$  is used to measure the degree of *consistency* of the evidence represented by  $\Gamma$  with the prior represented by  $\gamma$ . It follows that

$$\begin{aligned} \sum_{A \subseteq \Omega} m_M(A) &= \sum_{A \subseteq \Omega} Pr(\Gamma = A | \gamma \in \Gamma) \\ &= \sum_{A \subseteq \Omega} \frac{Pr(\Gamma = A \wedge \gamma \in \Gamma)}{Pr(\gamma \in \Gamma)} \\ &= \sum_{A \subseteq \Omega} \frac{Pr(\Gamma = A \wedge \gamma \in A)}{Pr(\gamma \in \Gamma)} \\ &= \sum_{A \subseteq \Omega} \frac{Pr(\Gamma = A)Pr(\gamma \in A)}{Pr(\gamma \in \Gamma)} \\ &= 1 \end{aligned}$$

So such a defined mapping  $m_M$  is actually a mass function on  $\Omega$  and is called *the induced mass function* of  $M$ .

Next we show that extended Dempster models are as expressive as standard Dempster models in the sense that any mass function  $m$  on  $\Omega$  can be represented as the induced mass function  $m_M$  of some extended Dempster model  $M$ . We prove a lemma which implies this expressiveness result.

**Lemma 3.5** For any mass function  $m$  and probability distribution  $pr$  on  $\Omega$ , there is an extended Dempster model  $M = \langle (U, Pr), \Gamma, \gamma, \Omega \rangle$  such that

1.  $m_M(A) = m(A)$  for each  $A \subseteq \Omega$  where  $m_M$  is the induced mass function of  $M$ ;
2.  $pr(A) = Pr(\gamma^{-1}(A))$  for any  $A \subseteq \Omega$ .

**Proof.** Given a mass function  $m$  and a probability function  $pr$  on  $\Omega$ , we define a mapping  $m_D : 2^\Omega \rightarrow [0, 1]$  as follows: for any  $A \subseteq \Omega$ ,

$$m_D(A) = \frac{\frac{m(A)}{pr(A)}}{\sum_{A \subseteq \Omega} \frac{m(A)}{pr(A)}} \quad (4)$$

Since  $\sum_{A \subseteq \Omega} m_D(A) = 1$ ,  $m_D$  is a mass function on  $\Omega$ . It follows that there is a standard Dempster model  $\langle (U_D, Pr_D), \Gamma_D, \Omega \rangle$  such that  $m_D(A) = Pr_D(\Gamma_D^{-1}(A))$  for any  $A \subseteq \Omega$ . From the proof of Lemma 3.4, we know that the extended Dempster model  $\langle (U_D, Pr_D), \Gamma_D, (\Omega, pr) \rangle$  with the prior probability  $pr$  can be represented as a Dempster model  $\langle (U, Pr), \Gamma, \Omega \rangle$  with  $\gamma$  as a mapping from  $U$  to  $\Omega$ . For this equivalent representation  $M := \langle (U, Pr), \Gamma, \gamma, \Omega \rangle$  of the extended model, we have that

- $Pr(\Gamma = A) = Pr(\Gamma^{-1}(A)) = Pr_D(\Gamma_D^{-1}(A)) = m_D(A)$ ;
- $Pr(\gamma \in A) = Pr(\gamma^{-1}(A)) = pr(A)$ .

It follows that

$$\begin{aligned} Pr(\gamma \in \Gamma) &= \sum_{A \subseteq \Omega} Pr(\Gamma = A) Pr(\gamma \in A) \\ &= \sum_{A \subseteq \Omega} m_D(A) pr(A) \\ &= \sum_{A \subseteq \Omega} \frac{\frac{m(A)}{pr(A)}}{\sum_{A \subseteq \Omega} \frac{m(A)}{pr(A)}} pr(A) \\ &= \frac{\sum_{A \subseteq \Omega} m(A)}{\sum_{A \subseteq \Omega} \frac{m(A)}{pr(A)}} \\ &= \frac{1}{\sum_{A \subseteq \Omega} \frac{m(A)}{pr(A)}} \end{aligned}$$

So we have that the induced mass function  $m_M$ :

$$\begin{aligned} m_M(A) &= \frac{Pr(\Gamma = A | \gamma \in \Gamma)}{Pr(\gamma \in \Gamma)} \\ &= \frac{Pr(\gamma \in A) Pr(\Gamma = A)}{Pr(\gamma \in \Gamma)} \\ &= m(A). \end{aligned}$$

QED

From the above proof, we know that, for any extended Dempster model  $M = \langle (U, Pr), \Gamma, (\Omega, pr) \rangle$ , there are two *induced* mass functions on  $\Omega$ : the induced mass

function  $m_D(A) (= Pr(\Gamma = A))$  in the part  $D := \langle (U, Pr), \Gamma, \Omega \rangle$ , which is actually a standard Dempster model, and the induced mass function  $m_M(A) (= Pr(\Gamma = A | \gamma \in \Gamma))$  of  $M$ .  $m_D$  measures the *belief update* and is called *basic certainty value*, while  $m_M$  measures *absolute belief*. This distinction is crucial to our following extension of Smets' transferable belief models with probabilistic priors. In our extended belief models, we use mass functions  $m_D$  for believing and mass functions  $m_M$  for decision-making. Mass functions for believing are based on the theory of evidence while mass functions for decision-making are essentially Bayesian and hence consistent with pignistic transformation. Basic certainty values are used in the probabilistic interpretation of CF in MYCIN [HECKERMAN, 1985]. For a given extended Dempster model  $M = \langle (U, Pr), \Gamma, (\Omega, pr) \rangle$ , there is a one-to-one correspondence (see Eqs.(3) and (4)) between the induced mass function  $m_M$  of  $M$  and the induced  $m_D$  in the standard-Dempster-model part  $D = \langle (U, Pr), \Gamma, \Omega \rangle$ . Assume that  $pr$  is given. The induced mass function  $m_M$  can be expressed in terms of  $m_D$  as follows:  $m_M(A) = \frac{pr(A)m_D(A)}{\sum_{A \subseteq \Omega} pr(A)m_D(A)}$ . We denote this expression as  $m_M = m_D \circ pr$ . Moreover,  $m_D$  can be expressed in terms of  $m_M$ :  $m_D(A) = \frac{m_M(A)/pr(A)}{\sum_{A \subseteq \Omega} m_M(A)/pr(A)}$ , which is denoted as  $m_D = m_M/pr$ . From the proof of Lemma 3.5, we know that the two operations  $\circ$  and  $/$  are reverse to each other in the sense that  $(m_D \circ pr)/pr = m_D$  and  $(m_M/pr) \circ pr = m_M$ .

Let  $M_1 = \langle (U_1, Pr_1), \Gamma_1, (\Omega, pr) \rangle$  and  $M_2 = \langle (U_2, Pr_2), \Gamma_2, (\Omega, pr) \rangle$  be two extended Dempster models representing two *independent* bodies of evidence on the same probability space  $(\Omega, pr)$ . Let  $m_{D_1}$  and  $m_{D_2}$  be the two induced mass functions for belief updates in the standard-Dempster-model parts  $D_1 = \langle (U_1, Pr_1), \Gamma_1, \Omega \rangle$  and  $D_2 = \langle (U_2, Pr_2), \Gamma_2, \Omega \rangle$ , respectively. As in Dempster models,  $m_{D_1}$  and  $m_{D_2}$  are combined according to the well-known *Dempster's rule*: for any  $C \subseteq \Omega$ ,

$$(m_{D_1} \oplus_D m_{D_2})(C) = \frac{\sum_{A_1 \cap A_2 = C} m_{D_1}(A_1) m_{D_2}(A_2)}{K_D} \quad (5)$$

where  $K_D = \sum_{A_1 \cap A_2 \neq \emptyset} m_{D_1}(A_1) \cdot m_{D_2}(A_2)$  is the normalization factor. So the combination  $(m_{D_1} \oplus_D m_{D_2})$  also measures belief update for the same probability space  $(\Omega, pr)$ . Let  $m_{M_1}$  and  $m_{M_2}$  denote the two induced mass functions for *absolute belief* on the extended Dempster models  $M_1$  and  $M_2$ , respectively. Now we provide a *new* combination rule for the extended Dempster models as follows: for any  $C \subseteq \Omega$ ,

$$\begin{aligned} &(m_{M_1} \oplus_M m_{M_2})(C) \\ &= \frac{\sum_{A_1 \cap A_2 = C} \frac{pr(A_1 \cap A_2)}{pr(A_1)Pr(A_2)} m_{M_1}(A_1) m_{M_2}(A_2)}{K_M} \end{aligned} \quad (6)$$



where

$$K_M := \sum_{A_1 \cap A_2 \neq \emptyset} \frac{pr(A_1 \cap A_2)}{pr(A_1)pr(A_2)} m_{M_1}(A_1) m_{M_2}(A_2)$$

is the normalization factor. The following proposition says that the new combination  $\oplus_M$  of mass functions for absolute beliefs is consistent with the Dempster combination  $\oplus_D$  of their corresponding mass functions for belief updates.

**Proposition 3.6** *The combination  $m_{M_1} \oplus_M m_{M_2}$  of  $m_{M_1}$  and  $m_{M_2}$  for absolute belief satisfies the following property:*

$$m_{M_1} \oplus_M m_{M_2} = (m_{D_1} \oplus_D m_{D_2}) \circ pr. \quad (7)$$

**Proof.** For any  $A \subseteq \Omega$ ,

$$\begin{aligned} & \sum_{A \subseteq \Omega} [pr(A) \sum_{A_1 \cap A_2 = A} m_{D_1}(A_1) m_{D_2}(A_2)] \\ = & \sum_{A \subseteq \Omega} [pr(A) \sum_{A_1 \cap A_2 = A} \frac{m_{M_1}(A_1)}{pr(A_1)} \frac{m_{M_2}(A_2)}{pr(A_2)}] \\ = & \sum_{A \subseteq \Omega} [\sum_{A_1 \cap A_2 = A} \frac{pr(A)}{pr(A_1)pr(A_2)} \frac{m_{M_1}(A_1)}{K_1} \frac{m_{M_2}(A_2)}{K_2}] \end{aligned}$$

where  $K_1 = \sum_{A \subseteq \Omega} \frac{m_{M_1}(A)}{pr(A)}$  and  $K_2 = \sum_{A \subseteq \Omega} \frac{m_{M_2}(A)}{pr(A)}$ . The first equality comes from Eq.(3) and the second from Eq.(4). So we have

$$\begin{aligned} & ((m_{D_1} \oplus_D m_{D_2}) \circ pr)(A) \\ = & \frac{(m_{D_1} \oplus_D m_{D_2})(A) pr(A)}{\sum_{A \subseteq \Omega} (m_{D_1} \oplus_D m_{D_2})(A) pr(A)} \\ = & \frac{pr(A) \sum_{A_1 \cap A_2 = A} m_{D_1}(A_1) m_{D_2}(A_2)}{\sum_{A \subseteq \Omega} [pr(A) \sum_{A_1 \cap A_2 = A} m_{D_1}(A_1) m_{D_2}(A_2)]} \\ = & \frac{\sum_{A_1 \cap A_2 = A} \frac{pr(A)}{pr(A_1)pr(A_2)} \frac{m_{M_1}(A_1)}{K_1} \frac{m_{M_2}(A_2)}{K_2}}{\sum_{A \subseteq \Omega} [\sum_{A_1 \cap A_2 = A} \frac{pr(A)}{pr(A_1)pr(A_2)} \frac{m_{M_1}(A_1)}{K_1} \frac{m_{M_2}(A_2)}{K_2}]} \\ = & \frac{\sum_{A_1 \cap A_2 = A} \frac{pr(A)}{pr(A_1)pr(A_2)} m_{M_1}(A_1) m_{M_2}(A_2)}{\sum_{A \subseteq \Omega} [\sum_{A_1 \cap A_2 = A} \frac{pr(A)}{pr(A_1)pr(A_2)} m_{M_1}(A_1) m_{M_2}(A_2)]} \\ = & (m_{M_1} \oplus_M m_{M_2})(A) \end{aligned}$$

QED

### 3.2 TWO CONDITIONING RULES

There are two types of conditioning in Bayesian probability theory [DUBOIS AND DENOEU, 2012]. The first one is known as *revision*. Given a probability function  $Pr$  (which usually is a subjective probability), one learns a hard evidence in terms of a sure event  $C$ . The problem is to determine the new subjective probability measure  $Pr'$ , such that  $Pr'(C) = 1$ , according to some minimal change principle. The other one is called *prediction*. When dealing with prediction, we have at our disposal a model of uncertainty in the form of a probability

measure  $Pr$  issued from a representative set of statistical data. Moreover, given the knowledge  $C$  on the current state of the world, we combine this knowledge with the belief model  $Pr$  and predict some property  $A$  of the current world with its associated degree of belief  $Pr(A|C)$ . For belief functions, however, these two types of conditioning are *essentially* different and the mainstream literature is a revision theory of handling singular uncertain evidence [SHAFER, 1976], not so much an extension of Bayesian statistical prediction, although Dempster's pioneering works on upper and lower probabilities are motivated by statistical reasoning. The well-known Dempster's rule of conditioning, which is a special case of Dempster's rule of combination, can be viewed as a revision process. In general, prediction cannot be achieved using Dempster conditioning [DUBOIS AND DENOEU, 2012]. Fagin and Halpern [FAGIN AND HALPERN, 1991] and Jaffray [JAFFRAY, 1992] provided two prediction-style conditioning rules which generalize Bayesian prediction by interpreting belief functions as inner measures and lower probabilities, respectively.

In this paper, we provide a *new* prediction-style conditioning rule which is *consistent* with the revision style of conditioning performed according to Dempster's rule of conditioning. For a given extended Dempster model  $M = \langle (U, Pr), \Gamma, (\Omega, pr) \rangle$ , the prediction conditioning is carried out for the induced mass function  $m_M$  while the revision rule is for the induced mass function  $m_D$  of the part  $D = \langle (U, Pr), \Gamma, \Omega \rangle$  without the prior probability  $pr$ . These two conditioning are consistent in the sense of Proposition 3.6 when the certain knowledge  $C$  is represented by a categorical mass function with  $C$  as its only focal set. Our following rule for prediction-style conditioning provides a formula of how to compute conditional belief  $m_M(\cdot|C)$  on the knowledge  $C$ . Generally, for each  $E \subseteq \Omega$ , we transfer a proportion  $r_E \cdot m_M(E)$ , where  $0 \leq r_E \leq 1$ , to  $E \cap C$  and  $(1 - r_E) \cdot m_M(E)$  to  $E \cap \bar{C}$  [DUBOIS AND DENOEU, 2012]. In particular, when  $E \cap C = \emptyset$ , we set  $r_E = 0$ , which contributes nothing to  $E \cap C$ ; when  $E \subseteq C$ , we set  $r_E = 1$  and leave the whole  $m_M(E)$  to  $E$ . According to this idea, we obtain a *general* formula for conditioning:

$$m(A|C) := \frac{\sum_{E \cap C = A} r_E \cdot m_M(E)}{\sum_{E \cap C \neq \emptyset} r_E \cdot m_M(E)}.$$

It is easy to check that

- $m(A|C)$  is exactly the Dempster rule of conditioning in the case when  $r_E = 1$  iff  $E \cap C \neq \emptyset$ ;
- $m(A|C)$  is exactly the geometric rule of conditioning in the case when  $r_E = 0$  iff  $E \not\subseteq C$ .

In this paper, we define a *new* rule for prediction-style conditioning between the above two by setting  $r_E = \frac{pr(E \cap C)}{pr(E)}$ :

$$m_M(A|C) = \frac{\sum_{E \cap C = A} \frac{pr(E \cap C)}{pr(E)} \cdot m_M(E)}{K}$$

where  $K = \sum_{E \cap C \neq \emptyset} \frac{pr(E \cap C)}{pr(E)} \cdot m_M(E)$  is a normalization factor. It is easy to see that the conditioning  $m_M(\cdot|C)$  is a special case of the new combination rule for absolute beliefs (Eq.(6)) when the knowledge  $C$  is represented by a categorical mass function with  $C$  as its only focal set.

**Example 3.7** (Continue with Example 3.2) Assume that the jailer's preference over possible choices according to  $a$ 's knowledge is represented by a uniform distribution  $pr$  on  $\Omega$ . We obtain the induced mass function  $m_M : m_M(\{(a, b), (a, c)\}) = 1/2$ ,  $m_M(b, c) = 1/4$ ,  $m_M(c, b) = 1/4$  and hence the corresponding beliefs  $Bel_M(E_a|J_b) = 1/2 = Pl_M(E_a|J_b)$ .

By using our definition of conditioning rule  $m_M(\cdot|C)$  on the *certain* knowledge  $C$ , we define its corresponding Jeffrey's rule when the prior knowledge is uncertain and is represented by a probability function  $pr_e$  on a coarsening of  $\Omega$ :  $(\Omega, \mathcal{B})$  where  $\mathcal{B}$  is a subalgebra of the powerset  $2^\Omega$  with its atoms forming a partition of  $\Omega$ . Let  $At(\mathcal{B})$  denote the set of atoms of  $\mathcal{B}$ . A mass function  $m'_M$  on  $(\Omega, 2^\Omega)$  is said to be obtained from  $m_M$  by *belief kinematics* on  $(\Omega, \mathcal{B})$  if, for any  $B \in At(\mathcal{B})$ ,

$$m_M(A|B) = m'_M(A|B) \text{ for all } A \subseteq \Omega. \quad (8)$$

$m'_M$  is called the mass function proposed by *Jeffrey's rule* if it is obtained as follows: for any  $A \subseteq \Omega$ ,

$$m'_M(A) = \sum_{B \in At(\mathcal{B})} m_M(A|B)pr_e(B), \quad (9)$$

Intuitively, the above principle of belief kinematics on  $(\Omega, \mathcal{B})$  says that, even though  $m_M$  and  $m'_M$  may disagree on propositions on  $(\Omega, \mathcal{B})$ , they agree on their relevance to every proposition  $A \subseteq \Omega$ .

## 4 EXTENDED TRANSFERABLE BELIEF MODELS WITH PROBABILISTIC PRIORS

**Definition 4.1** Let  $m_M$  be the induced mass function of an extended Dempster model  $\langle (U, Pr), \Gamma, (\Omega, pr) \rangle$ . Its associated *pignistic probability function*  $Betp_{m_M}$  on  $\Omega$  is defined as follows: for any  $A \subseteq \Omega$ ,

$$Betp_{m_M}(A) = \sum_{E \subseteq \Omega} m_M(E) \frac{pr(E \cap A)}{pr(E)} \quad (10)$$

The transformation between  $m_M$  and  $Betp_{m_M}$  is called *the generalized pignistic transformation*. When the context is clear, we simply call it pignistic transformation.  $\triangleleft$

Since  $m_M(A) = \frac{m_D(A)pr(A)}{\sum_{E \subseteq \Omega} m_D(E)pr(E)}$ , the pignistic probability function can be expressed in terms of the mass function  $m_D$  for belief updates:  $Betp_{m_M}(A) = \frac{\sum_{E \subseteq \Omega} m_D(E)pr(E \cap A)}{\sum_{E \subseteq \Omega} m_D(E)pr(E)}$ . Note that Smets' pignistic transformation is not a special case of the above defined generalized pignistic transformation when the prior probability  $pr$  is the uniform distribution on  $\Omega$ .

**Example 4.2** (Continue with Example 3.7) We may *complete* the above partial model  $\langle \Omega, m \rangle$  and obtain a probabilistic model according to the uniform distribution  $pr$ . When  $a$  is to be executed, the "chances" of the jailer's saying  $b$  or  $c$  are equal. So  $a$  will distribute the mass  $m(E_a)$  equally between  $(a, b)$  and  $(a, c)$ . Then we have  $m(b, c) = m(c, b) = 1/3$  and  $m(a, b) = m(a, c) = 1/6$ , which is exactly the probability function according to Smets' pignistic transformation. Also we obtain the corresponding beliefs  $Bel(E_a|J_b) = 1/3 = Pl(E_a|J_b)$ , which is the same as expected according to Bayesian reasoning. However, this distribution is not the same as the one obtained according to the above generalized pignistic transformation in Eq.(10). Instead,  $Betp_{m_M}(a, b) = Betp_{m_M}(a, c) = Betp_{m_M}(c, b) = Betp_{m_M}(b, c) = 1/4$ .

Assume that  $m_1, \dots, m_l$  are induced mass functions on  $(\Omega, pr)$  and  $p_1, \dots, p_l$  are non-negative numbers such that  $\sum_{i=1}^l p_i = 1$ . It is interesting to note that pignistic transformation  $Betp$  satisfies the following *linearity property*:

$$Betp\left(\sum_{i=1}^l p_i m_i\right) = \sum_{i=1}^l p_i Betp(m_i). \quad (11)$$

This property is both the major requirement that led Smets to the solution for the pignistic transformation [SMETS, 2005] and the crucial step to show the commutativity of the diagrams in the following Theorem 4.5. In addition to the linearity property, Smets proposed other requirements: credal-pignistic link, projectivity, continuity, efficiency, anonymity and impossible event [SMETS, 2005]. These requirements lead to the *unique* solution of Smets' pignistic transformation. One can check that our *generalized* pignistic transformation meets all these requirements except the anonymity one. The anonymity requirement rephrases a general form of insufficient reason principle and hence is equivalent to the constraint that the prior probability in the extended Dempster model is uniform.

**Definition 4.3** An *extended transferable belief model* (ETBM)  $\mathbf{M} = \langle M, Betp \rangle$  is a two level mental model: the credal level where beliefs are represented by an extended Dempster model  $M = \langle (U, Pr), \Gamma, (\Omega, pr) \rangle$ , and the pignistic level where the pignistic probability function is obtained from the induced mass function  $m_M$  of  $M$  by the generalized pignistic transformation  $Betp$ .  $\triangleleft$

Smets' transferable belief model is a special case of the above defined extended transferable belief model when the prior probability is uniform.

**Theorem 4.4** Let  $Cond^p$  and  $Cond$  denote the above defined prediction style conditioning operator for mass functions and the standard one for Bayesian probability functions, respectively. We have that the following diagram commutes:

$$\begin{array}{ccc} (m_M, C) & \xrightarrow{Cond^p} & m_M(\cdot|C) \\ \downarrow Betp & & \downarrow Betp \\ (Pr, C) & \xrightarrow{Cond} & Pr(\cdot|C) \end{array}$$

**Theorem 4.5** Let  $m_M, pr_e$  and  $m'_M$  be as in Eq.(9). Probability measures  $Pr$  and  $Pr'$  denote the pignistic probability functions of  $m_M$  and  $m'_M$ , respectively. Then the following diagram commutes:

$$\begin{array}{ccc} (m_M, pr_e) & \xrightarrow{J} & m'_M \\ \downarrow Betp & & \downarrow Betp \\ (Pr, pr_e) & \xrightarrow{J} & Pr' \end{array}$$

where the first  $J$  is the Jeffrey conditioning for mass functions as defined in Eq.(9) and the second  $J$  denotes the standard Jeffrey conditioning in Bayesian probability theory. In other words, our Jeffrey's rule is nothing but the linearity property in Eq.(11).

The above two theorems tell us that in extended transferable belief models the two new conditioning rules are consistent with pignistic transformation; in other words, the following two strategies are equivalent: we can revise the pignistic probabilities which are transformed from the prior beliefs with Bayes rule applied to the (certain or uncertain) knowledge, or revise the prior beliefs at the credal level by the above two conditioning rules and recompute the pignistic transformation.

However, from Example 1.1, we know that marginalization or coarsening is *inconsistent* with pignistic transformation. That is to say, pignistic transformation is sensitive to the choice of frame of discernment, which causes the partial dissociation between the credal and pignistic levels. In the remainder of this section, we show that, in an extended TB-M, these two levels can be coordinated by transforming its prior probability function.

Let  $M = \langle (U, Pr), \Gamma, (\Omega, pr) \rangle$  be a given extended Dempster model. Let  $m_M$  and  $m_D$  denote the induced mass functions for absolute beliefs and belief updates, respectively. Let  $(\Omega, \mathcal{B})$  be a coarsening of  $\Omega$  where  $\mathcal{B}$  is a subalgebra of the powerset of  $\Omega$  with its atoms  $\mathcal{C} :=$

$\{B_1, \dots, B_n\}$  forming a partition of  $\Omega$ . So each element of  $\mathcal{B}$  is a disjoint union of some atoms from the basis  $\mathcal{C}$ . Correspondingly, the coarsening  $\Gamma^C$  of the multivalued mapping  $\Gamma$  must be defined in the following way: for any  $u \in U$ ,  $\Gamma^C(u) = \mathbf{B}(\Gamma(u))$  where  $\mathbf{B}$  denotes the operation of taking upper approximation in the subalgebra  $\mathcal{B}$ . The natural associated prior probability function  $pr_0^C$  in the coarsening frame is given by  $pr_0^C(B) := pr(B)$  for all  $B \in \mathcal{B}$ . Consider the coarsened extended Dempster model  $M_0^C = \langle (U, Pr), \Gamma^C, ((\Omega, \mathcal{B}), pr_0^C) \rangle$ . It is easy to check that the associated belief function for belief update remains unchanged: for any  $B \in \mathcal{B}$ ,  $(Bel_D)_B(B) = Bel_D(B)$  where  $Bel_D$  and  $(Bel_D)_B$  are the belief functions corresponding to the mass functions  $m_D$  and  $(m_D)_B$ , respectively. But the pignistic probabilities may change:  $Betp_{m_{M_0^C}}(B) \neq Betp_{m_M}(B)$  for some  $B \in \mathcal{B}$ .

In order to coordinate pignistic probabilities with coarsening, we need to *transform* the prior probability function  $pr_0^C$  to a new prior probability  $pr^C$  on the coarsening  $(\Omega, \mathcal{B})$  such that the pignistic probabilities on the new coarsening frame  $M^C := \langle (U, Pr), \Gamma^C, ((\Omega, \mathcal{B}), pr^C) \rangle$  are the same as those on the original extended Dempster model  $M = \langle (U, Pr), \Gamma, (\Omega, pr) \rangle$ : for all  $B_i \in \mathcal{C}$ ,  $Betp_{m_M}(B_i) = Betp_{m_{M^C}}(B_i)$ . This equality is equivalent to the following one:

$$\frac{\sum_{B \in \mathcal{B}} (m_D)_B(B) pr^C(B \cap B_i)}{\sum_{B \in \mathcal{B}} (m_D)_B(B) pr^C(B)} = Betp_{m_M}(B_i). \quad (12)$$

Let  $Pl(B_i)$  denote the sum  $\sum_{B_i \subseteq B} (m_D)_B(B)$ , where  $1 \leq i \leq n$ . It is easy to see that  $\sum_{B \in \mathcal{B}} (m_D)_B(B) pr^C(B) = \sum_{1 \leq i \leq n} pr^C(B_i) Pl(B_i)$  and  $\sum_{B \in \mathcal{B}} (m_D)_B(B) pr^C(B \cap B_i) = pr^C(B_i) Pl(B_i)$ . So the equality (12) is reduced to the following form: for any  $1 \leq i \leq n$ ,

$$\frac{pr^C(B_i)}{\sum_{1 \leq i \leq n} pr^C(B_i) Pl(B_i)} = \frac{Betp_{m_M}(B_i)}{Pl(B_i)}. \quad (13)$$

In this equation,  $pr^C(B_i)$  is the only unknown quantity. Since there are  $n$  equations with  $n$  unknowns in the group  $G$  of Eq.(13), this group has at least one solution. But we don't know whether this solution is nonnegative or not. Now we provide a *constructive* solution to  $G$ . Let  $K$  denote  $\sum_{B \in \mathcal{B}} pr^C(B_i) Pl(B_i)$  and  $a_i = \frac{Betp_{m_M}(B_i)}{Pl(B_i)}$ . The above group of equations can be simplified as follows:  $pr^C(B_i) = a_i K$ ,  $1 \leq i \leq n$ . Since  $\sum_{1 \leq i \leq n} pr^C(B_i) = 1$ , we get the following equation by adding the equations in  $G$  together:  $1 = (a_1 + a_2 + \dots + a_n)K$ . So we get:  $K = \frac{1}{\sum_{1 \leq i \leq n} \frac{Betp_{m_M}(B_i)}{Pl(B_i)}}$ . Finally we solve  $G$  and obtain the following solutions: for any  $1 \leq i \leq n$ ,

$$pr^C(B_i) = \frac{\frac{Betp_{m_M}(B_i)}{Pl(B_i)}}{\sum_{1 \leq i \leq n} \frac{Betp_{m_M}(B_i)}{Pl(B_i)}}. \quad (14)$$

**Theorem 4.6** *The above defined coarsening frame  $M^C = \langle (U, Pr), \Gamma^C, ((\Omega, \mathcal{B}), pr^C) \rangle$  with the prior probability  $pr^C$  given in Eq.(14) is consistent with pignistic transformation. Let  $m_{m_{M^C}}$  be the induced mass function of  $M^C$  and  $m_{D^C}$  be the induced mass function of the Dempster part  $D^C := \langle (U, Pr), \Gamma^C, (\Omega, \mathcal{B}) \rangle$ . Then we have:*

- $(m_D)_B(B) = m_{D^C}(B)$ ;
- $Betp_{m_{M^C}}(B) = Betp_{m_M}(B)$  for all  $B \in \mathcal{B}$ .

So  $pr^C$  serves as a coordinator between believing represented by  $m_{D^C} (= (m_D)_B)$  and betting by  $Betp_{m_{M^C}}$  on  $M^C$  by recording the sensitivity of the pignistic probabilities derived from  $m_{D^C}$ . Pignistic transformation provides a credal-pignistic link (Assumption 3.1 in [SMETS, 2005]);  $pr^C$  here offers another credal-pignistic link between pignistic probabilities ( $Betp_{m_M}(B_i)$ ) and plausibility  $Pl(B_i)$  (defined in terms of  $m_D$ ) for belief update.

As for Example 1.1, according to the above formulation, we have that  $Betp_{m_M^D}(D_1) = \frac{1}{3}$  and  $Betp_{m_M^D}(\{D_2, D_3\}) = \frac{2}{3}$ . So, since pignistic probabilities are insensitive to the choice of frame,  $Betp_{m_M^C}(C_1) = \frac{1}{3}$  and  $Betp_{m_M^C}(C_2) = \frac{2}{3}$ . Moreover, we get that  $Pl(C_1) = 1$  and  $Pl(C_2) = 1$ . Finally we obtain the prior probability on the frame  $\Omega^C : pr^C(C_1) = \frac{1}{3}$  and  $pr^C(C_2) = \frac{2}{3}$ .

## 5 RELATED WORKS AND CONCLUSIONS

Yen ([YEN, 1986]) extended the multivalued mapping in the DS theory to a probabilistic one that uses conditional probabilities to express the uncertain associations. He also proposed a combination similar to our rule in Eq.(6) and discussed its relationship to Dempster's rule of combination. Moreover, he distinguished between mass functions for belief update and those for absolute beliefs. Such a distinction motivated our definition of generalized pignistic transformation in extended TBM. But his framework differs from ours in that Yen considered probabilistic multivalued mapping while our probabilistic extension is about prior knowledge base. Our method of combining evidence with prior knowledge is similar to [MAHLER, 1996, FIXSEN AND MAHLER, 1997]. Mahler proposed a similar combination rule and investigated its relationship with Bayesian parallel combination. More importantly, he pointed out the connection between his combination rule and pignistic transformation. He extended DS theory mainly from the perspective of random sets while we stick to the Dempster-model approach. Our work essentially differs from those papers in that we focus on both the partial dissociation of betting from believing and the (in)sensitivity of pignistic probabilities to the choice of frame of discernment. Wilson [WILSON, 1993] did study

the sensitivity problem of pignistic probabilities in TBM. But he stayed within the DS theory without considering any probabilistic extension.

In order to translate DS models into probability models which are consistent with belief-function semantics (especially Dempster's rule of combination), Cobb and Shenoy [COBB AND SHENOY, 2006] proposed another probability transformation method called *plausibility transformation* as an alternative to pignistic transformation. Plausibility transformation enjoys many interesting properties. The most important one is the so-called regularity property, i.e., plausibility transformation turns Dempster combination of belief functions into "pointwise" combination of probability functions. But, as Cobb and Shenoy [COBB AND SHENOY, 2006] pointed out, another important operation in DS belief networks, coarsening (or marginalization), is not invariant under this transformation. In fact there is no probability transformation for DS models with Dempster's rule of combination that enjoys the regularity property and makes coarsening invariant [COBB AND SHENOY, 2006]. For a more comprehensive survey of probability transformation, one may refer to [CUZZOLIN, 2015]. There are many proposals for Jeffrey's rule in DS theory [MA ET AL., 2010, MA ET AL., 2011, SMETS, 1993, ZHOU ET AL., 2014]. But none of these Jeffrey's rules was proposed from the perspective of pignistic transformation as in this paper. Our proposed conditioning rules are consistent with pignistic transformation.

In order to focus on pignistic transformation, we simplify the presentation in this paper by taking a closed world assumption, which is different from Smets' open world assumption for TBM. Moreover, here we choose to represent beliefs with Dempster models, which is opposed to Smets' TBM without probabilistic interpretation. So we would like to investigate the extension of TBM with probabilistic priors under the open-world assumption and its probabilistic interpretation.

### Acknowledgements

The first author is partly supported by Key project for basic research from the Ministry of Science and Technology of China (Grant No. 2012CB316205), NSF of China (Grant No. 61370053) and the RUC foundation (Grant No. 2012030005). The second author is supported by ARC Discovery Project (ARC DP130102764), NSF of China (Grant Nos. 61428208 and 61472412), AMSS-UTS Joint Research Laboratory for Quantum Computation, Chinese Academy of Sciences, and the CAS/SAFEA International Partnership Program for Creative Research Team.

## References

- [COBB AND SHENOY, 2006] COBB, B. AND SHENOY, P. (2006). ON THE PLAUSIBILITY TRANSFORMATION METHOD FOR TRANSLATING BELIEF FUNCTION MODELS TO PROBABILITY MODELS. *Int. J. Approx. Reasoning*, 41(3):314–330.
- [CUZZOLIN, 2015] CUZZOLIN, F. (2015). *Geometry of Uncertainty*. SPRINGER. TO APPEAR.
- [DUBOIS AND DENOEU, 2012] DUBOIS, D. AND DENOEU, T. (2012). CONDITIONING IN DEMPSTER-SHAFFER THEORY: PREDICTION VS. REVISION. IN DENOEU, T. AND MASSON, M.-H., EDITORS, *Belief Functions*, VOLUME 164 OF *Advances in Soft Computing*, PAGES 385–392. SPRINGER.
- [FAGIN AND HALPERN, 1991] FAGIN, R. AND HALPERN, J. (1991). A NEW APPROACH TO UPDATING BELIEFS. IN BONISSONE, P., M., H., L., K., AND J., L., EDITORS, *UAI*, PAGES 347–374. ELSEVIER.
- [FIXSEN AND MAHLER, 1997] FIXSEN, D. AND MAHLER, R. (1997). THE MODIFIED DEMPSTER-SHAFFER APPROACH TO CLASSIFICATION. *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, 27(1):96–104.
- [GRÜN WALD AND HALPERN, 2003] GRÜN WALD, P. AND HALPERN, J. (2003). UPDATING PROBABILITIES. *J. Artif. Intell. Res. (JAIR)*, 19:243–278.
- [HALPERN, 2005] HALPERN, J. (2005). *Reasoning about Uncertainty*. MIT PRESS.
- [HECKERMAN, 1985] HECKERMAN, D. (1985). PROBABILISTIC INTERPRETATION FOR MYCIN’S CERTAINTY FACTORS. IN KANAL, L. N. AND LEMMER, J. F., EDITORS, *UAI ’85: Proceedings of the First Annual Conference on Uncertainty in Artificial Intelligence, Los Angeles, CA, USA, July 10-12, 1985*, PAGES 167–196. ELSEVIER.
- [JAFFRAY, 1992] JAFFRAY, J. (1992). BAYESIAN UPDATING AND BELIEF FUNCTIONS. *IEEE Transactions on Systems, Man, and Cybernetics*, 22(5):1144–1152.
- [MA ET AL., 2010] MA, J., LIU, W., DUBOIS, D., AND PRADE, H. (2010). REVISION RULES IN THE THEORY OF EVIDENCE. IN *ICTAI (1)*, PAGES 295–302. IEEE COMPUTER SOCIETY.
- [MA ET AL., 2011] MA, J., LIU, W., DUBOIS, D., AND PRADE, H. (2011). BRIDGING JEFFREY’S RULE, AGM REVISION AND DEMPSTER CONDITIONING IN THE THEORY OF EVIDENCE. *International J. on AI Tools*, 20 (4):691–720.
- [MAHLER, 1996] MAHLER, R. (1996). COMBINING AMBIGUOUS EVIDENCE WITH RESPECT TO AMBIGUOUS A PRIORI KNOWLEDGE. I. BOOLEAN LOGIC. *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, 26(1):27–41.
- [SHAFFER, 1976] SHAFFER, G. (1976). *A Mathematical Theory of Evidence*. PRINCETON UNIVERSITY PRESS, PRINCETON, N.J.
- [SMETS, 1993] SMETS, P. (1993). JEFFREY’S RULE OF CONDITIONING GENERALIZED TO BELIEF FUNCTIONS. IN HECKERMAN, D. AND MAMDANI, E. H., EDITORS, *UAI*, PAGES 500–505. MORGAN KAUFMANN.
- [SMETS, 2005] SMETS, P. (2005). DECISION MAKING IN THE TBM: THE NECESSITY OF THE PIGNISTIC TRANSFORMATION. *Int. J. Approx. Reasoning*, 38(2):133–147.
- [SMETS AND KENNES, 1994] SMETS, P. AND KENNES, R. (1994). THE TRANSFERABLE BELIEF MODEL. *Artif. Intell.*, 66(2):191–234.
- [SNOW, 1998] SNOW, P. (1998). THE VULNERABILITY OF THE TRANSFERABLE BELIEF MODEL TO DUTCH BOOKS. *Artif. Intell.*, 105(1-2):345–354.
- [WILSON, 1993] WILSON, N. (1993). DECISION-MAKING WITH BELIEF FUNCTIONS AND PIGNISTIC PROBABILITIES. IN CLARKE, M., KRUSE, R., AND MORAL, S., EDITORS, *ECSCQARU’93, Granada, Spain, November 8-10, 1993, Proceedings*, VOLUME 747 OF *Lecture Notes in Computer Science*, PAGES 364–371. SPRINGER.
- [YEN, 1986] YEN, J. (1986). A REASONING MODEL BASED ON AN EXTENDED DEMPSTER-SHAFFER THEORY. IN KEHLER, T., EDITOR, *UAI, 1986. Volume 1: Science.*, PAGES 125–131. MORGAN KAUFMANN.
- [ZHOU ET AL., 2014] ZHOU, C., WANG, M., AND QIN, B. (2014). BELIEF-KINEMATICS JEFFREY’S RULES IN THE THEORY OF EVIDENCE. IN ZHANG, N. AND TIAN, J., EDITORS, *UAI 2014*, PAGES 917–926. AUAI PRESS.

---

# Probabilistic Graphical Models Parameter Learning with Transferred Prior and Constraints

---

Yun Zhou<sup>†,‡</sup>, Norman Fenton<sup>†</sup>, Timothy M. Hospedales<sup>†</sup>, Martin Neil<sup>†</sup>

<sup>†</sup> Risk and Information Management (RIM) Research Group, Queen Mary University of London

<sup>‡</sup> Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology

## Abstract

Learning accurate Bayesian networks (BNs) is a key challenge in real-world applications, especially when training data are hard to acquire. Two approaches have been used to address this challenge: 1) introducing expert judgements and 2) transferring knowledge from related domains. This is the first paper to present a generic framework that combines both approaches to improve BN parameter learning. This framework is built upon an extended multinomial parameter learning model, that itself is an auxiliary BN. It serves to integrate both knowledge transfer and expert constraints. Experimental results demonstrate improved accuracy of the new method on a variety of benchmark BNs, showing its potential to benefit many real-world problems.

## 1 INTRODUCTION

Directed probabilistic graphical models, also known as Bayesian networks (BNs), are a natural framework for describing probabilistic dependencies among variables in many real-world problems, such as medical symptom diagnosis (Velikova et al., 2014) and software defect prediction (Fenton and Neil, 2014). However, in problem domains with limited or no relevant training data, there are major challenges in accurately learning BN parameters (Friedman et al., 1999).

There are several methods for handling parameter learning with limited or no relevant data, described in a rich literature of books, articles and software packages, which are briefly summarized in (Druzdel and Van Der Gaag, 2000; Neapolitan, 2004; O’Hagan et al., 2006). Without considering any domain knowledge, the simplest learning approaches usually fail to accurately estimate parameters in a small dataset. To mitigate this problem, it may be possible to elicit numerical assessments from expert judgements, but this process is inefficient and error-prone.

Researchers have shown that experts tend to feel more comfortable providing qualitative or semi-numerical judgments (Feelders and van der Gaag, 2006) with less cognitive effort. Such judgments expressed as constraints between parameters of interest (e.g. “the probability of people getting cancer is smaller than 1%”) are more reliable than numerical assessments, and have drawn considerable attention recently. In the work of (Zhou et al., 2014a), these kinds of constraints are modelled as nodes in an auxiliary BN model called MPL-C (Multinomial Parameter Learning model with Constraints), which includes nodes modelling training data statistics. The MPL-C improves parameter estimation accuracy by constraining the estimation with the expert constraints.

An alternative approach to improving BN learning in scarce data situations is to transfer knowledge from different but related BNs that may have more training data available (Luis et al., 2010). For example, transferring knowledge from the same medical diagnosis network learned in a different country. This can be effective if data for one or more sufficiently related source domains is available. However, the practical limitation is that transfer is contingent on availability of suitable related sources, and the relatedness of each source to the target task may not be known in advance. Estimating relatedness is thus important but challenging in practice, particularly when there are multiple potential sources of possibly varying relatedness.

While incorporating either parameter constraints or transfer learning from related data in source domains can improve parameter estimation accuracy, there exists no generic learning framework to synergistically exploit the benefits of both approaches. Achieving this is non-trivial because typical approaches to transfer (Luis et al., 2010) and to constrained learning (Zhou et al., 2014a) use very different formalisations. In this paper we generalise the state-of-the-art MPL-C model for learning with expert constraints to also exploit knowledge from related source domains via a bootstrap approach. The new model called MPL-TC (Multinomial Parameter Learning model with Transferred prior and Constraints) synergistically exploits both forms of external

knowledge to improve learning performance in a target BN.

The rest of the paper is organized as follows. Section 2 discusses the related work. Section 3 provides a formalisation of the BN parameter learning problem. Section 4 introduces the MPL-C model and shows how it can be used to learn with parameter constraints. Section 5 describes our novel generalisation for constrained parameter learning with transfer from auxiliary sources (MPL-TC). Our method estimates relatedness to pick the best source to transfer from, and takes a bootstrap approach for generating target parameter priors from the source data. Section 6 gives empirical results on a set of benchmark datasets. Section 7 concludes the paper and discusses the future work.

## 2 RELATED WORK

Several models have been proposed to integrate parameter constraints and improve learning accuracy. The constrained convex optimization (CO) formulation (Altendorf et al., 2005; Niculescu et al., 2006; de Campos and Ji, 2008; de Campos et al., 2008; Liao and Ji, 2009; de Campos et al., 2009; Yang and Natarajan, 2013) is the most popular way to estimate the constrained parameters. In this setting, the algorithm seeks the globally optimal estimation (maximal data log-likelihood) with respect to the parameter constraints. The parameters also can be estimated by Monte Carlo methods (Chang et al., 2008), where only the samples that satisfy the constraints are kept. Recently, auxiliary BN models (Zhou et al., 2014a,b) have been developed for solving this problem. This approach provides an extensible framework for parameter learning with additional information. However, these auxiliary models only use uniform parameter priors to regularize learning, which can be improved by informative priors (Neapolitan, 2004).

In the context of transfer learning in BNs, the multi-task framework of (Niculescu-mizil and Caruana, 2007) considers structure transfer. However, it assumes that all sources are equally related and simply learns the parameters for each task independently. The transfer framework of (Luis et al., 2010) (referred to as CPTAgg in this paper) measures the relatedness of tasks via calculating K-L divergence between target and source CPTs, and employs the heuristic weighted sum model for aggregating target and selected source parameters. The weights are proportional to the number of training samples. Finally, the study (Oyen and Lane, 2012) considers multi-task structure learning, again with independently learned parameters. Their model shows transfer performs poorly without knowledge of relatedness. However, they address this by using manually specified relatedness.

No previous work considers a generic BN parameter learning framework combining both transferred knowledge and constraints as discussed in this paper. Using the bootstrap approach for variability measurement in BNs is not new.

For example, Friedman et al. (1999) study the robustness of network features based on DAGs learned on bootstrap resamples. Elidan (2011) uses bootstrap aggregation (bagging) to find a stable prediction model through improved computation of the log-likelihood score. However, to the best of our knowledge, this is the first work to use bootstrap to measure the variability of source MLEs and generate *TNormal*<sup>1</sup> parameter priors for transfer purpose.

## 3 BACKGROUND

A Bayesian network (BN) consists of three components: variables  $V = \{X_1, X_2, X_3, \dots, X_n\}$  corresponding to nodes of the BN, a set of numerical parameters  $\theta$  of the variables in  $V$ , and a Directed Acyclic Graph (DAG)  $G$  encoding the statistical dependencies among the variables. For discrete variables, the probability distribution is described by a conditional probability table (CPT) that contains the probability of each value of the variable given each instantiation of its parents as defined by graph  $G$ . We write this as  $p(X_i|\pi_i)$  where  $\pi_i$  denotes the set of parents of variable  $X_i$  in DAG  $G$ . Thus, the BN defines a simplified joint probability distribution over  $V$  given by:

$$p(X_1, X_2, \dots, X_n) = \prod_{i=1}^n p(X_i|\pi_i) \quad (1)$$

Let  $r_i$  denote the cardinality of the space of  $X_i$ , and  $|\pi_i|$  represent the cardinality of the space of parent configurations of  $X_i$ . The  $k$ -th probability value of a conditional probability distribution  $p(X_i|\pi_i = j)$  can be represented as  $\theta_{ijk} = p(X_i = k|\pi_i = j)$ , where  $\theta_{ijk} \in \theta$ ,  $1 \leq i \leq n$ ,  $1 \leq j \leq |\pi_i|$  and  $1 \leq k \leq r_i$ .

In our BN parameter learning setting, we have data  $D$  combined with  $V$  and  $G$  to form the problem domain  $\mathcal{D} = \{V, G, D\}$ . Within a domain  $\mathcal{D}$ , the goal of parameter learning is to determine parameters for all  $p(X_i|\pi_i)$ . Given data  $D$ , the estimation of CPT parameters  $\theta$  is conventionally solved by the Maximum Likelihood Estimation (MLE),  $\hat{\theta} = \arg \max_{\theta} \log p(D|\theta)$ . Let  $N_{ijk}$  be the number of data samples in  $D$  for which  $X_i$  takes its  $k$ -th value and its parents set  $\pi_i$  takes its  $j$ -th value, and  $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$ . The MLE estimate for each parameter is:

$$\hat{\theta}_{ijk} = \frac{N_{ijk}}{N_{ij}} \quad (2)$$

However, it is common (even for large datasets) that certain parent-child state combinations seldom appear, and MLE learning fails in this situation. Another classic parameter learning algorithm (Maximum a Posteriori, MAP) mitigates this by introducing a Dirichlet prior on  $\theta$  so that:  $\hat{\theta} = \arg \max_{\theta} \log p(D|\theta)p(\theta)$ . This results in the MAP estimate  $\hat{\theta}_{ijk} = \frac{N_{ijk} + \alpha_{ijk}}{N_{ij} + \alpha_{ij}}$ . Intuitively, the hyperparameters  $\alpha_{ijk}$  in the Dirichlet prior correspond to an expert's

<sup>1</sup>The abbreviation of *Truncated Normal* distribution.

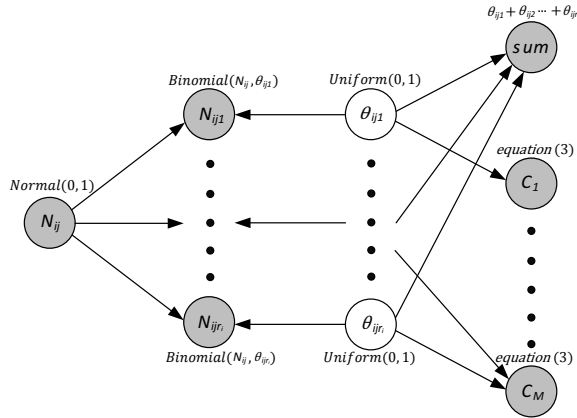
guess of the corresponding virtual data counts. When there is no expert judgment, the K2 ( $\alpha_{ijk} = 1$ ) or BDeu<sup>2</sup> ( $\alpha_{ijk} = \frac{1}{|\pi_i| r_i}, \forall i, j, k$ ) priors are commonly used (Heckerman et al., 1995).

## 4 MPL-C MODEL

For parameter learning with constraints, the auxiliary multinomial parameter model has been proposed (MPL-C (Zhou et al., 2014a)). This method treats the parameters of interest and constraints as nodes in an auxiliary model. The parameter learning process is achieved via auxiliary model inference given observed data statistics and constraint conditions.

### 4.1 MODEL CONSTRUCTION

We use the estimation of probability distribution  $p(X_i | \pi_i = j)$  (i.e., the  $j$ -th column<sup>3</sup> of the CPT associated with the variable  $X_i$ ) as an example to illustrate the construction of the MPL-C model. Suppose there are  $r_i$  states, and the goal is to learn the  $r_i$  probability parameters  $\theta_{ij1}, \dots, \theta_{ijr_i}$  corresponding to these states. Assume we have  $N_{ijk}$  data observations of the  $k$ -th state ( $1 \leq k \leq r_i$ ) and the total number of observations is  $N_{ij}$ . Then we can create a multinomial parameter learning BN model for each CPT column (shown schematically in Figure 1) to estimate parameters  $\theta_{ij1}, \dots, \theta_{ijr_i}$ .



**Figure 1:** Graphical model representation of MPL-C with  $M$  constraints. For the constraint nodes, their equations follow the representations in equation 3. The gray nodes are observed during the inference.

Specifically, we start by creating an integer node named  $N_{ijk}$  (corresponding to  $N_{ijk}$  as defined above) for each  $k$  that is *Binomial* distributed. This node has two parents

<sup>2</sup>Bayesian Dirichlet likelihood equivalent uniform prior.

<sup>3</sup>Note in some other works, e.g., Netica BN software, each CPT row represents a discrete probability distribution given a parent configuration.

$N_{ij}$  and  $\theta_{ijk}$  to model the total number of trials and success probability in the *Binomial* distribution. The  $N_{ij}$  has a *Normal* distribution<sup>4</sup>, which provides an infinite range for the total number of trials. The prior distribution of each  $\theta_{ijk}$  is uniform between 0 and 1. Finally, there is an integer node *sum*, which is a shared child of  $\theta_{ijk}$  ( $k = 1, \dots, r_i$ ). This node models the normalization constraint for the all success probabilities, i.e., that they should sum to 1.

### 4.2 INCORPORATING CONSTRAINTS

In real-world applications, many expert judgments can be described with linear inequality constraints and approximate equality constraints (Zhou et al., 2014a) having the following generic form:

$$\begin{cases} \beta_0 + \sum_{k=1}^{r_i} \beta_k \theta_{ijk} \leq 0 \\ |\theta_{ijk} - \theta_{ijk'}| \leq \varepsilon \quad (0 < \varepsilon < 1) \end{cases} \quad (3)$$

Here the coefficients  $\beta_0, \beta_k$  ( $1 \leq k \leq r_i$ ) are real numbers, and  $\varepsilon$  is an appropriate (small) positive value selected by the expert to represent  $\theta_{ijk} \approx \theta_{ijk'}$ .

Given that an expert has identified a number of constraints as defined above within a CPT column, then these constraints can be integrated as additional observed constraint nodes within the MPL model to generate a new model called MPL-C as shown in Figure 1.

Each constraint node  $C_m$  is a deterministic binary (*true/false*) node with expressions that specify the constraint relationships between its parents:

$$\begin{aligned} & \text{if}(\beta_0 + \sum_{k=1}^{r_i} \beta_k \theta_{ijk} \leq 0, \text{true}, \text{false}) \\ & \text{if}(\text{abs}(\theta_{ijk} - \theta_{ijk'}) \leq \varepsilon, \text{true}, \text{false}) \end{aligned}$$

When the constraint is between a single parameter and a constant (i.e.,  $\beta_0 + \beta_k \theta_{ijk} \leq 0$ ), the constraint node will only have a single parent. Inference for the unobserved  $\theta_{ijk}$  in this auxiliary model implements constrained MAP parameter learning for one CPT column of the target BN. In the next section we show how to generalise this framework to also take into account knowledge transfer from related source domains.

## 5 MPL-C MODEL WITH TRANSFERRED PRIOR

### 5.1 THE TRANSFER LEARNING MODEL

The idea behind transfer learning is to improve the accuracy of a target BN by making use of one or more related source BNs. For example, the target BN may be a model

<sup>4</sup>This can be replaced with *Poisson* distribution to only allow positive integers. Because this root node is always observed with a valid number of trials during the inference, using *Normal* or *Poisson* distribution will produce the same results.



for diagnosis of a particular disease based on limited data in one district or country. If there are other (source) BNs with similar variables and objectives but from a different district or country, then it make sense to exploit such models to improve the accuracy of the target BN. Transfer learning does this by providing methods for both determining suitability of the data in the source and its transfer to the target.

The obvious practical limitation of transfer learning – which limits the applicability of all work in this area including this paper – is that the relatedness is never truly known. The necessary assumptions to overcome this introduce inevitable bias into the results. In this paper we assume there is at least one source domain that is sampled from similar distributions as the target, and that this can be transferred to help learn the target BN parameters. However determining relatedness in a data driven way means there is an inevitable confirmation bias in the sense that the source BNs most likely to be selected are those that most closely match the current target estimate. This limits the extent that the source can ‘change’ the target when it is more ‘correct’ than the current noisy target estimates.

If the chosen source and target are not sampled from similar distributions, directly applying parameters learned in another domain may be impossible or result in negative transfer: the underlying tasks may have major quantitative or qualitative differences (e.g., care procedures vary across hospitals). This limits the effectiveness of existing methods such as CPTAgg in (Luis et al., 2010). Our framework will address this by robustly measuring piecewise relatedness.

Given the scarce data for parameter estimation in the target domain  $\mathcal{D}^t = \{V^t, G^t, D^t\}$ , the knowledge in available source domains should be mined to help the learning in the target domain. We aim to learn a target domain  $\mathcal{D}^t$  leveraging sources  $\{\mathcal{D}^s\}$  with potentially *piecewise* relatedness. Typically relatedness is computed at domain or instance level granularity, but for more flexible transfer we model relevance as varying within-domain. Thus transfer can still be exploited when different subsets of features/variables are relevant to different source domains. Thus we allow the heterogeneity  $V^t \neq V^s$  and  $G^t \neq G^s$ , and transfer at the level of BN *fragments*.

**Definition 1 BN fragment.** A Bayesian network of domain  $\mathcal{D}$  can be divided into a set of sub-networks (denoted *fragments*)  $\mathcal{D} = \{\mathcal{D}_i\}$  by considering the graph  $G$ . Each fragment  $\mathcal{D}_i = \{V_i, G_i, D_i\}$  is a single root node or a node with its direct parents in the original BN, and encodes a single CPT from the original BN. The number of fragments is the number of variables in the original BN.

To achieve flexible BN parameter transfer, the target domain and source domains are all broken into fragments  $\mathcal{D}^t = \{\mathcal{D}_i^t\}$ ,  $\{\mathcal{D}^s\} = \{\{\mathcal{D}_{i'}^s\}\}$ . Assuming for now no latent variables in the target domain, then each target fragment  $i$  can be learned independently  $\hat{\theta}_i^t =$

$\arg \max_{\theta_i^t} p(\theta_i^t | C_i^t, \mathcal{D}_i^t, \{\{\mathcal{D}_{i'}^s\}\})$ . To leverage the bag of source domain fragments  $\{\{\mathcal{D}_{i'}^s\}\}$  in learning each  $\theta_i^t$ , we consider each source fragment  $\mathcal{D}_{i'}^s$  as potentially relevant. Specifically, for each target fragment, every source fragment is evaluated for relatedness and the best fragment mapping is chosen. Once the best source fragment is chosen for each target, it will be used as parameter priors in the target MPL-C model. This auxiliary model can then be updated to infer the parameter posteriors given target data  $D_i^t$ , target constraints  $C_i^t$  and source networks  $\{\mathcal{D}^s\}$ .

To realize this strategy, three issues must be addressed: 1) which source fragments are transferrable, 2) how to deal with variable name mapping, 3) how to quantify the relatedness of each transferrable source fragment in order to find the best one. We next address each of these issues in turn:

**Fragment Compatibility** For a target fragment  $i$  and putative source fragment  $i'$ , we say they are *compatible* if they have the same structure and state space. That is, the same number of states and parents states. Additionally, if the target fragment contains parameter constraints  $C_i^t$ , the associated source parameter  $\theta_{i'}^s$  should fall in the constrained value ranges  $\Omega_{C_i^t}$ , so

$$compatible(\mathcal{D}_i^t, \mathcal{D}_{i'}^s) = \begin{cases} 1 & \text{if } G_i^t = G_{i'}^s \ \& \ \theta_{i'}^s \in \Omega_{C_i^t} \\ & \ \& \ dim(\theta_i^t) = dim(\theta_{i'}^s) \\ 0 & \text{otherwise} \end{cases}$$

where  $dim(\theta_i^t) = dim(\theta_{i'}^s)$  means  $r_i^t = r_{i'}^s$  and  $|\pi_i^t| = |\pi_{i'}^s|$ . This assumption could be relaxed quite straightforwardly at the expense of additional computational cost. For example, if the target variable contains 2 states (*true* and *false*), and one source variable contains 3 states (*high*, *medium* and *low*), we can try multiple aggregations of the source states to generate three mappings to the target: 1) *true* – *high* and *false* – *medium, low*; 2) *true* – *medium* and *false* – *high, low*; 3) *true* – *low* and *false* – *high, medium*.

**Fragment Permutation Mapping** For two fragments  $i$  and  $i'$  determined to be compatible, we still may not know the mapping between variable names. For example, if  $i$  has parents  $[a, b]$  and  $i'$  has parents  $[d, c]$ , the correspondence could be  $a - d, b - c$  or  $b - d, a - c$ . The function *permutations*( $G_i^t, G_{i'}^s$ ) returns an exhaustive list of possible mappings  $P_m$  that map states of  $i'$  to states of  $i$ .

**Fitness Measurement** To measure the relatedness between compatible target and source fragments  $\mathcal{D}_i^t$  and  $\mathcal{D}_{i'}^s$ , we use Bayesian model comparison for two hypotheses:  $H_1$  is the relevance hypothesis<sup>5</sup> that the source and target data share a common CPT, and  $H_0$  (not  $H_1$ ) is the indepen-

<sup>5</sup>Simplifying the fragment notation, so  $H_1$  only refers the dependent hypothesis between  $\mathcal{D}_i^t$  and  $\mathcal{D}_{i'}^s$ .

dent hypothesis that the source and target data have distinct CPTs. These two hypotheses are the outputs of our function  $fitness(\mathcal{D}_i^t, \mathcal{D}_{i'}^s, p(H))$ , and can be computed with:

$$\begin{aligned} p(H_1|D_{i'}^s, D_i^t) &\propto \int p(D_i^t|\theta_i)p(\theta_i|D_{i'}^s, H_1)p(H_1)d\theta_i, \\ p(H_0|D_{i'}^s, D_i^t) &\propto \int p(D_i^t|\theta_i^t)p(\theta_i^t|H_0)p(H_0)d\theta_i^t. \end{aligned} \quad (4)$$

For discrete data, the likelihood of  $H_1$ , integrating out the unknown CPTs  $\theta_i$ , is the Dirichlet compound multinomial (DCM) or *Pólya* distribution:

$$p(D_i^t|D_{i'}^s, H_1) = \sum_{j=1}^{|\pi_i|} \left( \frac{\Gamma(\alpha_{i'j}^s)}{\Gamma(N_{ij}^t + \alpha_{i'j}^s)} \prod_{k=1}^{r_i} \frac{\Gamma(N_{ijk}^t + \alpha_{i'jk}^s)}{\Gamma(\alpha_{i'jk}^s)} \right) \quad (5)$$

where  $\alpha_{i'jk}^s$  indicates the aggregate counts from the source domain and distribution prior, and  $\alpha_{i'j}^s = \sum_k \alpha_{i'jk}^s$ . Maximising  $p(H_1|D_{i'}^s, D_i^t)$  over source networks  $s$  and fragment  $i'$  finds the fragment most likely to share the same generating distribution as the target, and thus the best source to transfer. Next we will discuss how to use the selected source data to help learn the target parameters.

## 5.2 THE MPL-TC MODEL

Given a target fragment and selected source fragment, the challenge is to fuse them in a robust manner. We solve this fusion problem in a Bayesian way – treating the transferred information as the target parameter priors. We refer to this framework as MPL-TC, which contains three main steps: 1) for each BN fragment in the target domain, find its closest source fragment and permutation in the source domain; 2) transfer the selected source fragment by converting the source data statistics into prior distributions of parameters in the target MPL-C model (see Section 4) and 3) perform the inference in this auxiliary model to learn the target parameters. The detail can be found in Algorithm 1.

**Fragment Fusion** To fuse the selected source fragment with the target fragment in the second step of our algorithm, we perform the bootstrap approach in the source to generate the priors of target parameters. Bootstrap is a resampling method to measure the quality of true samples (Duval, 1993). In this paper, we are interested in the quality of selected source parameters. We cannot access infinite training samples of selected source, instead we only have a sample of it (the selected best mapping source sample  $D_{i'j}^s$ ), which means the MLE of  $\theta_{i'jk}^s$  is not accurate.

From a specific subset of source samples, i.e.,  $D_{i'j}^s$ , only one estimate of the MLE for a parameter of interest  $\theta_{i'jk}^s$  can be obtained. In order to reason about the population, we need some sense of the variability of the estimated MLE. Thus, we apply the simplest bootstrap method – sam-

**INPUT** : Target domain  $\mathcal{D}^t$ , source domains  $\{\mathcal{D}^s\}$  and target constraints  $C^t$ .

**OUTPUT**: The target parameters  $\hat{\theta}^t$ .

**for each target fragment  $i$  do**

**for each source network  $s$  and fragment  $i'$  do**

**if compatible**( $\mathcal{D}_i^t, \mathcal{D}_{i'}^s$ ) **then**

$P = \text{permutations}(G_{i'}^t, G_{i'}^s)$ ;

**for permutation  $m = 1$  to  $M$  do**

Measure relatedness:

$fitness(\mathcal{D}_i^t, P_m(\mathcal{D}_{i'}^s)) =$

$p(H_1|D_i^t, P_m(D_{i'}^s))$

**end**

**end**

**end**

Find the best source fragment and permutation:

$\arg \max_{i', s, m} p(H_1|D_i^t, P_m(D_{i'}^s))$

**for each parent state configuration  $j$  do**

**for each state value  $k$  do**

$\{\theta_{i'jk}^s\} =$

$\text{bootstrap}(100, @MLE, P_m(D_{i'j}^s))$

Fit the  $\{\theta_{i'jk}^s\}$  with

$\zeta_{i'jk}^s = TNormal(\mu_{ijk}, \sigma_{ijk}, 0, 1)$

**end**

Generate the auxiliary model in the target:

$\Psi_{ij}^t = \text{mpltc}(D_{ij}^t, C_{ij}^t, \zeta_{i'jk}^s)$

Inference to get the parameters estimation:

$\hat{\theta}_{ij}^t = \text{inference}(\Psi_{ij}^t)$

**end**

**end**

**return**  $\hat{\theta}^t = \{\hat{\theta}_{ij}^t\}$

**Algorithm 1:** Multinomial Parameter Learning with Transferred Prior and Constraints

pling from the  $D_{i'j}^s$  to form a new sample (called a “resample” or bootstrap sample) that is also of size  $|D_{i'j}^s|$ . The bootstrap sample is taken from the original using sampling with replacement. This process is repeated multiple times (100 or 1000), and for each of these bootstrap samples we compute the MLE of  $\theta_{i'jk}^s$  (each of these are called bootstrap estimates). We now have a set of bootstrap estimates, which are used to fit a *TNormal* distribution to encode how much the source MLE varies.

In our MPL-TC approach, these *TNormal* distributions ( $\{\zeta_{i'jk}^s\}$ ) are used to replace the uniform parameter priors on  $\theta_{i'jk}^s$  (Figure 1) of MPL-C models in the target domain. Thus the transferred prior, target training samples and constraints are now all encoded in the target MPL-TC models (referred to as  $\Psi_{ij}^t$  in Algorithm 1). After observing the sources, the target data statistics ( $N_{ij}^t, N_{ij1}^t, \dots, N_{ijr_i}^t$ ) and available constraints (The constraint nodes are all observed

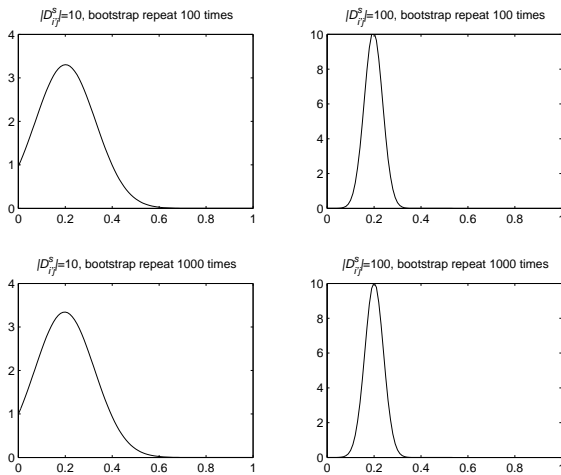
with ‘true’ values), we can update (by *inference*( $\cdot$ ) function in Algorithm 1) these auxiliary models to get the target parameter posteriors:

$$p(\hat{\theta}_{ij1}^t, \dots, \hat{\theta}_{ijr_t}^t | N_{ij}^t, N_{ij1}^t, \dots, N_{ijr_t}^t, \dots, C_1^t, \dots, C_M^t, \zeta_{i'jk}^s, \dots, \zeta_{i'jr_t}^s, sum)$$

Because the auxiliary BNs are hybrid models, the update/inference is performed via a state-of-the-art dynamic discretization junction tree (DDJT) algorithm (Neil et al., 2007) that is implemented in AgenaRisk<sup>6</sup>. The time complexity of the inference is exponential in model treewidth, which restricts the applicability at some point. However, approximate inference could be used with dynamic discretization to improve the time efficiency.

### 5.3 ILLUSTRATIVE EXAMPLES

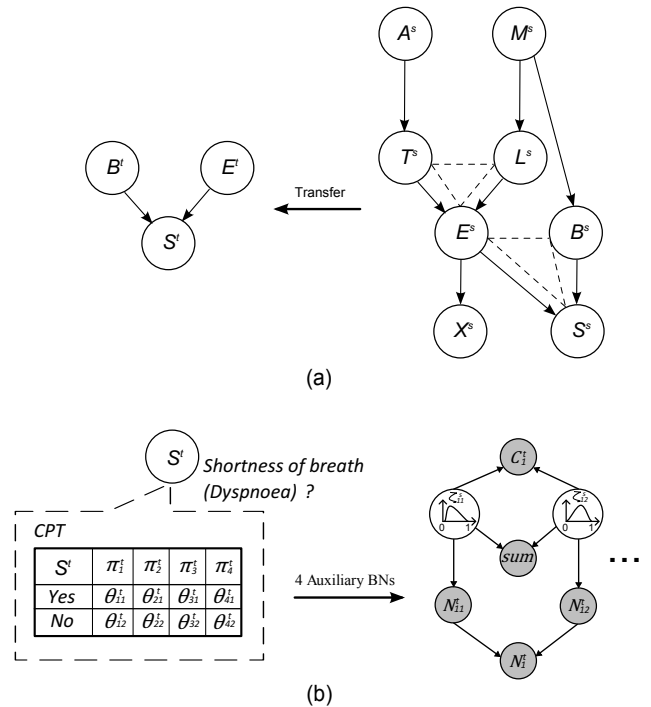
**Bootstrap Fitting of TNormal Priors** Figure 2 demonstrates an example of fitted *TNormal* distributions for MLE estimation  $\theta_{i'jk}^s = 0.2$  learned from sample sizes 10 and 100. As we can see, although the parameter estimations of two source samples are the same, the estimation from the large sample are definitely more reliable than the estimation from the small sample, where the fitted *TNormal* distribution in  $|D_{i'j}^s| = 100$  is much sharper than the distribution in  $|D_{i'j}^s| = 10$ . Moreover, the number of bootstrap replicates does not change the results much, and we use 100 replicates in all subsequent experiments.



**Figure 2:** The fitted *TNormal* distributions for a parameter of interest with different source sample sizes 10 and 100. The original source MLE estimation of this parameter is 0.2, which means there are 2 and 20 appearances of this parameter in sample sizes 10 and 100 respectively. The bootstrap process in each case is repeated 100 and 1000 times.

**Fragment Transfer** Here we provide an illustrative example of our framework for fragment-based parameter transfer and the target parameter estimation: the target is a three node BN shown in the left part of Figure 3(a), and

the source is an eight node BN shown in the right part of Figure 3(a). We aim to estimate the CPT of  $S^t$ , which has four parent state configurations –  $\pi_1^t, \pi_2^t, \pi_3^t$  and  $\pi_4^t$ . As we can see, there are two source fragments ( $\{T^s, L^s, E^s\}$  and  $\{E^s, B^s, S^s\}$ ) which are *compatible* with target fragment (shown with dashed triangle in Figure 3(a)). Thus, there are four *permutations* of compatible source fragments (assuming binary parent nodes). All four of these options are then evaluated for *fitness*, and the best fragment and permutation is picked ( $\{B^s, E^s, S^s\}$ ). In Figure 3(b), we generate four auxiliary BNs (MPL-TC model) for each target parameter column, the right part of Figure 3(b) shows the MPL-TC model of the first parameter column, which is used to estimate  $\theta_{11}^t$  and  $\theta_{12}^t$ . The constraints and data statistics in the target domain are modelled by nodes with gray color. The parameter priors (nodes with white color) are *TNormal* distributions fitted from source bootstrap samples. Finally, these priors are updated by observing the target data statistics and constraints to get the posterior parameter estimates.



**Figure 3:** A simple example to show the framework of multinomial parameter learning with transferred prior and constraints. (a) The dashed triangle represents source fragments  $\{T^s, L^s, E^s\}$  and  $\{E^s, B^s, S^s\}$ , which are *compatible* to the target fragment. (b) The structure representation of the MPL-TC model for estimating  $\theta_{11}^t$  and  $\theta_{12}^t$  is in the first target parameter column, whose parameter priors ( $\theta_{11}^s$  and  $\theta_{12}^s$ ) are converted from the most fit source fragment  $\{B^s, E^s, S^s\}$  via bootstrap.

## 6 EXPERIMENTS

### 6.1 EXPERIMENT SETTING

In all cases, we assume that the structure of the model is known and that the ‘true’ CPTs that we are trying to learn

<sup>6</sup><http://www.agenarisk.com/>

are those that are provided as standard with benchmark BN models. For the purpose of the experiment we are not given these true CPTs but instead are given a limited number of sample observations which are randomly generated based on the true target CPTs. To introduce noise between the target and source for simulating varying relatedness, the source datasets are also sampled from the true CPTs but with ‘soft’ and ‘hard’ noise conditions: (1) soft: generate three source domains with 200, 300 and 400 sample sizes to simulate continuously varying relatedness among a set of sources; (2) hard: choose a portion (20%) of each source’s fragments uniformly at random and randomise their data/CPTs to make them irrelevant. This results in a different subset of compatible but (un)related fragments in each source. Introducing these two types of sampling noise makes the sources similar but different to the target, and hence simulates the kind of source-target relations that may exist in practice.

The constraints are elicited from the true CPTs (so they are certainly correct) and randomly assigned to parameters in the network. Following the method of constraints generation in (Liao and Ji, 2009), for each true parameter  $\theta_{ijk}$ , we create a constraint:

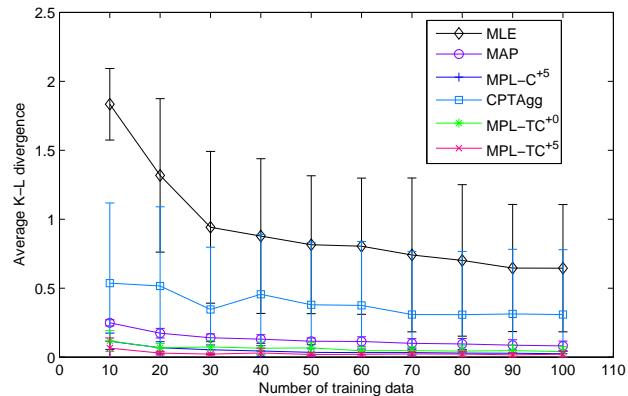
$$\min((1 + \varepsilon)\theta_{ijk}, 1) \geq \theta_{ijk}^t \geq \max((1 - \varepsilon)\theta_{ijk}, 0)$$

where the  $\varepsilon = 0.05$  in the experiments. These elicited constraints are encoded in the MPL-TC auxiliary models, which are built with BN software AgenaRisk.

We compare our MPL-TC<sup>+5</sup> against following algorithms and settings (the upper right superscript value associated with learning algorithms represents the number of constraints used in these algorithms):

- MLE and MAP, conventional BN parameter learning algorithms.
- MPL-C<sup>+5</sup>, state-of-the-art parameter learning algorithm with five constraints (Zhou et al., 2014a).
- CPTAgg, state-of-the-art parameter transfer learning algorithm (Luis et al., 2010).
- MPL-TC<sup>+0</sup>, our MPL-TC algorithm with zero constraints.

The resulting learned CPTs are evaluated against the true CPTs by using the K-L divergence measure (Kullback and Leibler, 1951). The smaller the K-L divergence is, the closer the estimated CPT is to the true CPT. Here the K-L divergence is locally measured for each CPT column and averaged over the whole model. This is to ensure that the fit of each distribution is equally weighted in the overall metric. Each experiment is repeated 10 times, and the results are reported with the mean and standard deviation of the K-L divergences between estimated and true CPTs.



**Figure 4:** Parameter learning performance in the Cancer BN under different levels of data sparsity. Lower is better.

## 6.2 EXPERIMENTS ON CANCER BN

The Cancer BN (Korb and Nicholson, 2010) models the interaction between risk factors and symptoms for the purpose of diagnosing the most likely condition for a patient getting lung cancer. This BN contains 5 *Boolean* nodes, so each CPT column has just 2 parameters to learn; since the parameters sum to 1, each column has only one independent parameter. Hence there are 10 independent parameters to learn in the model. In the target domain, training samples under different sparsity levels (10 to 100 samples) are drawn from the ground-truth Cancer BN.

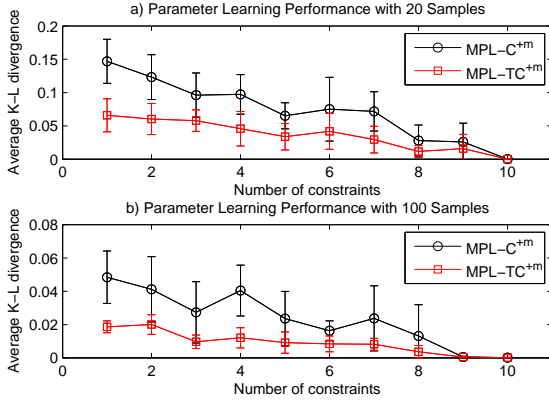
**Overall** Figure 4 presents the results of all learning algorithms under varying data volumes in the target Cancer BN. It is clear that the average K-L divergence of all learning algorithms decreases with increasing target sample size. With increasing sample sizes, the performance gap between the algorithms decreases<sup>7</sup>. Moreover, our MPL-TC<sup>+5</sup> always outperforms all the other competitors, which demonstrates the effectiveness of our framework.

Considering models without parameter constraints (MLE, CPTAgg, MAP and MPL-TC<sup>+0</sup>): MPL-TC<sup>+0</sup> provides overall K-L divergence reductions (performance improvements) of 93.4%, 84.1% and 52.3% compared with MLE, CPTAgg and MAP respectively, thus demonstrating the efficacy of knowledge transfer.

After introducing 5 sampled constraints, MPL-TC<sup>+5</sup> achieves even greater reductions in comparison with MLE, CPTAgg and MAP, which are 97.1%, 93.0% and 79.1% respectively. Due to the benefit of introducing parameter constraints, MPL-C<sup>+5</sup> algorithm also outperforms MLE, CPTAgg and MAP. However, MPL-TC<sup>+5</sup> still outperforms MPL-C<sup>+5</sup> with 42.0% average K-L divergence reduction.

According to the results, the MPL-TC<sup>+5</sup> greatly outperforms the conventional MLE and MAP algorithms, and the

<sup>7</sup>Given enough target training samples, the learning performance of all algorithms converge.



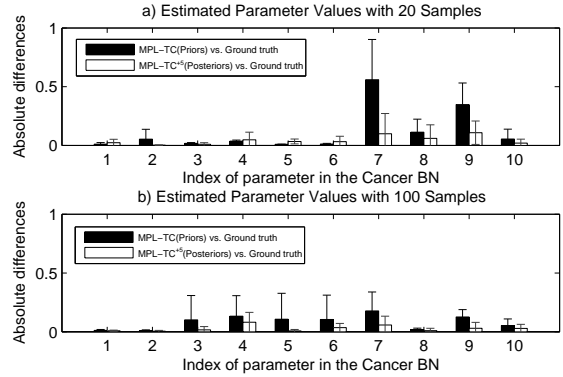
**Figure 5:** Performance of MPL-C and MPL-TC when varying the number of constraints ( $m = 1, \dots, 10$ ).

CPTAgg and MPL-C<sup>+5</sup> that only use transfer or constraints alone. This demonstrates the complementarity of both constraints and sources of external knowledge when learning with scarce target data.

**Varying Number of Constraints** To investigate how the number of introduced constraints affect the learning performance of MPL-C and MPL-TC, we vary the number of sampled constraints in parameter learning (shown in Figure 5). As we can see, K-L divergence decreases with more constraints for both MPL-C and MPL-TC in both data sparsity settings. However, when the number of constraints is small, our MPL-TC greatly outperforms MPL-C due to the benefit of transferred parameter priors. When the number of constraints increases to 10 (every parameter is constrained), the learning results of MPL-C and MPL-TC both converge to zero in both settings.

**Priors vs. Posteriors** To provide insight into the mechanisms of our framework, we investigate the differences between MPL-TC(Priors) (transferred  $TNormal$  mean values) and MPL-TC<sup>+5</sup>(Posteriors) (the updated parameter posteriors after inference given the target data and parameter constraints) for each parameter in the Cancer BN. The results are presented in Figure 6, where the heights of the bars represent the absolute differences between estimated values and true CPT values.

As we can see, the MPL-TC(Priors) shows inaccurate transfer in both two settings: parameters 7–9 in Figure 6(a) and parameters 3–7 and 9 in Figure 6(b). This is caused by the bias in target samples and noise in source domains. Therefore, the estimates of MPL-TC(Priors) are far from the true values, (average K-L divergence of 0.65 and 0.40 for sample sizes 20 and 100 respectively). However, after performing MAP learning in the MPL-TC<sup>+5</sup> model, the MPL-TC<sup>+5</sup>(Posteriors) reduces the average K-L divergence between the estimated values and true values to 0.09 and 0.03 respectively. These results demonstrate the robustness of the Bayesian learning in MPL-TC<sup>+5</sup>, and the importance of systematically inferring the new parameters



**Figure 6:** The differences between estimated probability values (MPL-TC(Priors) and MPL-TC<sup>+5</sup>(Posteriors)) and ground truth for all parameters in the Cancer BN.

given available data and constraints. Next, we will compare the performance of all these algorithms in different BN parameter learning problems.

### 6.3 EXPERIMENTS ON STANDARD BNS

We evaluate the algorithms on 12 standard BNs<sup>8</sup> (details in Table 1). For each BN, 100 training samples and 5 constraints are drawn from the true CPTs in the target domain.

**Overall** Table 1 summarises the average K-L divergence per parameter. The best results are presented in bold. The statistically significant improvements of the best result over competitors are indicated with asterisks \* (two-sample t-test at the default 5% significance level). In summary, the MPL-C<sup>+5</sup> and MPL-TC<sup>+5</sup> methods outperform conventional MLE and MAP in 11 out of 12 settings, the only exception is the learning performance in Weather BN, where the learning results of these methods converge with enough training samples<sup>9</sup>. These results demonstrate the benefit of learning with both sources of external knowledge. Compared with the state-of-the-art MPL-C<sup>+5</sup>, MPL-TC<sup>+5</sup> wins in every setting (including the Weather BN, where MPL-TC<sup>+5</sup> achieves even smaller average K-L divergence – 0.018 of MPL-TC<sup>+5</sup> vs. 0.020 of MPL-C<sup>+5</sup>). Over all BNs, MPL-TC<sup>+5</sup> gets 83.2%, 33.5% and 26.9% average reduction of K-L divergence compared with MLE, MAP and MPL-C<sup>+5</sup> respectively.

**Transfer vs. No Transfer** Considering transfer learning only, both CPTAgg and MPL-TC<sup>+0</sup> outperform conventional MLE, which demonstrate the benefit of introducing source domain knowledge. However, due to a simplistic relatedness model and CPT fusion heuristic, CPTAgg even fails to outperform MAP in some settings. In contrast, our MPL-TC<sup>+0</sup> outperforms CPTAgg and MAP with 74.1%

<sup>8</sup><http://www.bnlearn.com/bnrepository/>

<sup>9</sup>As shown in Table 1, the Weather BN only contains 9 parameters to learn, therefore 100 training samples are already enough to train a good model.

**Table 1:** Parameter learning performance (average K-L divergence) in 12 standard Bayesian networks.

| Name       | Nodes | Edges | Para | MLE               | MAP        | MPL-C <sup>+5</sup> | CPTAgg            | MPL-TC <sup>+0</sup> | MPL-TC <sup>+5</sup> |
|------------|-------|-------|------|-------------------|------------|---------------------|-------------------|----------------------|----------------------|
| Alarm      | 37    | 46    | 509  | 2.36±0.10*        | 0.66±0.01* | 0.61±0.02*          | 1.61±0.08*        | <b>0.42</b> ±0.02    | <b>0.42</b> ±0.01    |
| Andes      | 223   | 338   | 1157 | 1.03±0.06*        | 0.17±0.01* | 0.15±0.01*          | 0.65±0.05*        | <b>0.08</b> ±0.00    | <b>0.08</b> ±0.00    |
| Asia       | 8     | 8     | 18   | 0.57±0.16*        | 0.34±0.04* | 0.28±0.03*          | 0.31±0.05*        | 0.22±0.02*           | <b>0.18</b> ±0.03    |
| Cancer     | 5     | 4     | 10   | 0.86±0.35*        | 0.09±0.04* | 0.07±0.05*          | 0.54±0.11*        | 0.05±0.01*           | <b>0.03</b> ±0.01    |
| Earthquake | 5     | 4     | 10   | 1.50±0.82*        | 0.15±0.04* | 0.13±0.03*          | 0.35±0.22*        | 0.11±0.01            | <b>0.10</b> ±0.01    |
| Hailfinder | 56    | 66    | 2656 | 2.85±0.01*        | 0.46±0.00* | 0.41±0.00*          | 1.98±0.01*        | <b>0.31</b> ±0.01    | <b>0.31</b> ±0.01    |
| Hepar2     | 70    | 123   | 1453 | 3.18±0.13*        | 0.33±0.01* | 0.33±0.01*          | 2.58±0.15*        | 0.30±0.01            | <b>0.29</b> ±0.00    |
| Insurance  | 27    | 52    | 984  | 1.95±0.18*        | 1.17±0.03* | 1.07±0.03*          | 0.93±0.06*        | <b>0.75</b> ±0.03    | <b>0.75</b> ±0.02    |
| Sachs      | 11    | 17    | 178  | 1.74±0.29*        | 0.78±0.04* | 0.71±0.05*          | 0.98±0.08*        | <b>0.50</b> ±0.03    | <b>0.50</b> ±0.02    |
| Survey     | 6     | 6     | 21   | 0.35±0.20*        | 0.05±0.01* | 0.05±0.01*          | 0.24±0.15*        | 0.04±0.01            | <b>0.03</b> ±0.01    |
| Weather    | 4     | 4     | 9    | <b>0.02</b> ±0.02 | 0.03±0.00  | <b>0.02</b> ±0.00   | <b>0.02</b> ±0.00 | <b>0.02</b> ±0.00    | <b>0.02</b> ±0.00    |
| Win95pts   | 76    | 112   | 574  | 3.59±0.07*        | 0.81±0.01* | 0.78±0.02*          | 3.20±0.10*        | 0.67±0.02*           | <b>0.64</b> ±0.01    |

and 31.2% average reduction of K-L divergence over all the settings. In addition, after introducing both transferred parameter priors and target constraints, our MPL-TC<sup>+5</sup> shows additional improved learning performance over CPTAgg and MPL-TC<sup>+0</sup> (75.0% and 3.5% average reduction of K-L divergence).

**Importance of Transfer vs. Constraints** As we can see, our MPL-TC<sup>+0</sup> outperforms MPL-C<sup>+5</sup>, which indicates the transferred prior is more helpful than a moderate number (i.e., 5) of constraints in improving parameter learning performance in these experiments. Given the burden of constraint elicitation in the real world, we used a realistic limited number of constraints. Of course if sufficient constraints were available, MPL-C would perform better (cf Figure 5) and this result would be reversed. But in this case, the transferred prior makes a greater contribution in improving performance – despite the noise process between source and target domain, and the imperfect estimation of relevance. This is especially in the larger BNs, where the constraints are scarcer relative to the number of parameters to learn. This also explains why MPL-TC<sup>+0</sup> and MPL-TC<sup>+5</sup> have similar results in the Alarm, Andes, Hailfinder, Insurance and Sachs BNs.

## 7 DISCUSSION AND CONCLUSIONS

When data is scarce, purely data driven BN parameter learning is inaccurate. The broad goal of this paper was to introduce a new method (MPL-TC) that is the first attempt at BN parameter learning incorporating both transfer learning and qualitative constraints in a complementary way. Using the public BN repository, we showed that learning performance was greatly improved in MPL-TC across a range of networks. In particular, we demonstrated that MPL-TC worked well in every data and constraint sparsity in the Cancer BN, and achieved the best performance in all BNs in the repository compared with other state-of-the-art algorithms.

We currently assume there is at least one relevant source. For each target fragment, we find the most relevant source fragment to generate target parameter priors. Transferring to a target fragment using information from >1 sources would be a straightforward modification of the current framework. However it would increase the risk of ‘negative transfer’ (Torrey and Shavlik, 2009) that could be detrimental to performance (if some apparently relevant sources used for transfer are actually false positives). This trade off between maximum exploitable transfer, and robustness to negative transfer is pervasive in transfer learning.

We discussed the limitations of all BN transfer learning approaches with respect to the fact that, in practice relatedness is hard to guarantee or estimate. Thus data-driven transfer (source selection) may be biased by inaccurate target data (resulting in bad choice of source and thus negative transfer) in extremely scarce settings. In the spirit of synergistically combining source data and constraints, available target constraints clearly provide an opportunity to guide and disambiguate transfer. In this paper, we only used target parameter constraints to exclude individual incompatible source fragments. Richer models for guiding transfer with constraints including cross-node constraints, source-domain constraints, and cross-domain constraints should be investigated in future.

## 8 ACKNOWLEDGMENTS

The authors would like to thank three anonymous reviewers for their valuable feedback. This work is supported by the European Research Council (ERC-2013-AdG339182-BAYES-KNOWLEDGE) and the European Union’s Horizon 2020 research and innovation programme under grant agreement No 640891. YZ is supported by China Scholarship Council (CSC)/Queen Mary Joint PhD scholarships and National Natural Science Foundation of China (61273322, 71471174).

## References

- Altendorf, E.E., Restificar, A.C., Dietterich, T.G., 2005. Learning from sparse data by exploiting monotonicity constraints, in: Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence, pp. 18–26.
- de Campos, C.P., Ji, Q., 2008. Improving Bayesian network parameter learning using constraints, in: Proceedings of the 19th International Conference on Pattern Recognition, pp. 1–4.
- de Campos, C.P., Tong, Y., Ji, Q., 2008. Constrained maximum likelihood learning of Bayesian networks for facial action recognition, in: Proceedings of the 10th European Conference on Computer Vision. Springer, pp. 168–181.
- de Campos, C.P., Zeng, Z., Ji, Q., 2009. Structure learning of Bayesian networks using constraints, in: Proceedings of the 26th Annual International Conference on Machine Learning, ACM. pp. 113–120.
- Chang, R., Stetter, M., Brauer, W., 2008. Quantitative inference by qualitative semantic knowledge mining with Bayesian model averaging. Knowledge and Data Engineering, IEEE Transactions on 20, 1587–1600.
- Druzel, M., Van Der Gaag, L.C., 2000. Building probabilistic networks: “where do the numbers come from?”. IEEE Transactions on knowledge and data engineering 12, 481–486.
- Duval, R., 1993. Bootstrapping: A nonparametric approach to statistical inference. 94–95, Sage.
- Elidan, G., 2011. Bagged structure learning of Bayesian network, in: Proceedings of the 14th International Conference on Artificial Intelligence and Statistics, pp. 251–259.
- Feelders, A., van der Gaag, L., 2006. Learning Bayesian network parameters under order constraints. International Journal of Approximate Reasoning 42, 37–53.
- Fenton, N.E., Neil, M., 2014. Decision support software for probabilistic risk assessment using Bayesian networks. IEEE software 31, 21–26.
- Friedman, N., Goldszmidt, M., Wyner, A., 1999. Data analysis with Bayesian networks: A bootstrap approach, in: Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence, Morgan Kaufmann Publishers Inc.. pp. 196–205.
- Heckerman, D., Geiger, D., Chickering, D.M., 1995. Learning Bayesian networks: The combination of knowledge and statistical data. Machine Learning 20, 197–243.
- Korb, K.B., Nicholson, A.E., 2010. Bayesian Artificial Intelligence. CRC Press, New York.
- Kullback, S., Leibler, R.A., 1951. On information and sufficiency. The Annals of Mathematical Statistics , 79–86.
- Liao, W., Ji, Q., 2009. Learning Bayesian network parameters under incomplete data with domain knowledge. Pattern Recognition 42, 3046–3056.
- Luis, R., Sucar, L.E., Morales, E.F., 2010. Inductive transfer for learning Bayesian networks. Machine learning 79, 227–255.
- Neapolitan, R.E., 2004. Learning Bayesian networks. Pearson Prentice Hall.
- Neil, M., Taylor, M., Marquez, D., 2007. Inference in hybrid Bayesian networks using dynamic discretization. Statistics and Computing 17, 219–233.
- Niculescu, R.S., Mitchell, T., Rao, B., 2006. Bayesian network learning with parameter constraints. The Journal of Machine Learning Research 7, 1357–1383.
- Niculescu-mizil, A., Caruana, R., 2007. Inductive transfer for Bayesian network structure learning, in: Proceedings of the 11th International Conference on Artificial Intelligence and Statistics, pp. 1–8.
- O’Hagan, A., Buck, C.E., Daneshkhah, A., Eiser, J.R., Garthwaite, P.H., Jenkinson, D.J., Oakley, J.E., Rakow, T., 2006. Uncertain judgements: eliciting experts’ probabilities. Wiley.com.
- Oyen, D., Lane, T., 2012. Leveraging domain knowledge in multitask Bayesian network structure learning, in: Proceedings of the 26th AAAI Conference on Artificial Intelligence, pp. 1091–1097.
- Torrey, L., Shavlik, J., 2009. Transfer learning. Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques 1, 242.
- Velikova, M., van Scheltinga, J.T., Lucas, P.J., Spaanderman, M., 2014. Exploiting causal functional relationships in Bayesian network modelling for personalised healthcare. International Journal of Approximate Reasoning 55, 59–73.
- Yang, S., Natarajan, S., 2013. Knowledge intensive learning: Combining qualitative constraints with causal independence for parameter learning in probabilistic models, in: Machine Learning and Knowledge Discovery in Databases. Springer, pp. 580–595.
- Zhou, Y., Fenton, N., Neil, M., 2014a. Bayesian network approach to multinomial parameter learning using data and expert judgments. International Journal of Approximate Reasoning 55, 1252 – 1268.
- Zhou, Y., Fenton, N., Neil, M., 2014b. An extended MPL-C model for Bayesian network parameter learning with exterior constraints, in: van der Gaag, L., Feelders, A. (Eds.), Probabilistic Graphical Models. Springer International Publishing. volume 8754 of *Lecture Notes in Computer Science*, pp. 581–596.

**AUAI Press**  
**P.O. Box 866**  
**Corvallis, Oregon 97339**  
**USA**