Pacific Symposium on Biocomputing 11:547-558(2006)

CLASS PREDICTION FROM TIME SERIES GENE EXPRESSION PROFILES USING DYNAMICAL SYSTEMS KERNELS

KARSTEN M. BORGWARDT
Institute for Computer Science, Ludwig-Maximilians-University of Munich, Oettingenstr. 67, 80538 Munich, Germany
kb@dbs.ifi.lmu.de

S.V.N. VISHWANATHAN
Statistical Machine Learning Program, National ICT Australia, Canberra, 0200 ACT, Australia
SVN.Vishwanathan@nicta.com.au

HANS-PETER KRIEGEL
Institute for Computer Science, Ludwig-Maximilians-University of Munich, Oettingenstr. 67, 80538 Munich, Germany
kriegel@dbs.ifi.lmu.de

We present a kernel-based approach to the classification of time series of gene expression profiles. Our method takes into account the dynamic evolution over time as well as the temporal characteristics of the data. More specifically, we model the evolution of the gene expression profiles as a Linear Time Invariant (LTI) dynamical system and estimate its model parameters. A kernel on dynamical systems is then used to classify these time series. We successfully test our approach on a published dataset to predict response to drug therapy in Multiple Sclerosis patients. For pharmacogenomics, our method offers a huge potential for advanced computational tools in disease diagnosis, and disease and drug therapy outcome prognosis.

1 Introduction

Gene expression levels change over time, as proteins interfere with gene transcription. Proteins and DNA interact in a complex feedback system of gene expression control, in which some proteins foster gene expression as *transcription factors*, while others reduce transcription activity as *inhibitors* (for details see [1]). Furthermore, protein-protein interactions can increase or reduce the influence of certain proteins on transcription. These networks of gene expression control form the basis of essential cellular processes such as the cell cycle, development, and disease progression.

Single microarray profiles describe one current state of a cell only and may prove inadequate to study these complex interactions that steer biological processes. Therefore, it becomes necessary to view and analyze gene expression

profiles as dynamical systems evolving with time. Such an approach may lead to a more expressive model for interpreting molecular processes within cells.

Over recent years, a growing number of time series of microarray data has become available in databases such as the Stanford Microarray Database [2] and GEO [5]. Whereas early studies used algorithms on microarray time series that had been developed for static data (for example [12]), the interest in algorithms that are able to handle and analyze time series of gene expression data in particular has grown tremendously since. The interested reader is referred to [3] for a full review of approaches and challenges in this field.

1.1 Classification Using Time Series Gene Expression Profiles

Most data mining algorithms that have been developed for gene expression time series (for example [10]) deal with the problem of clustering. Given a set of data points D, clustering algorithms try to decompose D into subsets $\{D_1, \ldots, D_n\}$ such that similarity between data points is maximized within each cluster, i.e. each subset, and minimized between distinct clusters. In short, clustering finds classes of data when classes are unknown. In medical applications, clustering is most required when exploring subtypes of the same disease or when searching groups of genes with similar expression profiles, i.e. to find classes in unorganized data.

On the other hand, classification deals with the problem of predicting class membership of unlabeled test data points after learning from a training set of data points with known class memberships. Predicting class labels can be regarded as equivalent to predicting unknown characteristics of data points. Central questions in pharmacogenomics constitute such classification problems: Will patient X respond well to a certain therapy or drug treatment? Has patient X been infected by a pathogen? Is patient X recovering from a disease? Pharmacogenomics could greatly benefit from computational tools that answer or at least help human experts to answer these questions. The growing number of time series microarray data provide the training data from which such advanced classifiers can learn.

Goal and outline of this article In this project, our aim was as follows: Define a novel approach to time series microarray classification which uses Support Vector Machine (SVM) classification and a kernel function which respects the temporally changing character of these expression profiles. In what follows, we will present our dynamical systems kernel for gene expression time series data. Modeling the time series as dynamical systems, we measure distances between these systems and then classify them using a SVM. In Section 2, we

will briefly review kernel methods. In Section 3, we show how microarray time series data can be modeled as dynamical systems and how kernels can be defined on them. We show the feasibility of our approach in experiments in Section 4 on drug response prediction. We conclude with a discussion of our findings and an outlook to future extensions and refinements of our method.

2 Support Vector Machines and Kernels

In this section we give a brief overview of binary classification with SVMs and kernels. For a more extensive treatment we refer the reader to [11], and the references therein.

Given m observations (x_i, y_i) drawn iid (independently and identically distributed) from a distribution over $\mathcal{X} \times \{\pm 1\}$ our goal is to find a function $f : \mathcal{X} \to \{\pm 1\}$ which classifies observations $x \in \mathcal{X}$ into classes $+1$ and -1. In particular, SVMs assume that f is a linear function given by

$$f(x) = \text{sign}(\langle w, x \rangle + b), \tag{1}$$

and maximize the margin of separation between the decision boundary and the points from opposite classes. We also need to take into account the slack when the two classes are not linearly separable. Without going into details (which can be found in [11]) this leads to the optimization problem:

$$\begin{array}{ll} \underset{w,b,\xi}{\text{minimize}} & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m} \xi_i \\ \text{subject to} & y_i\left(\langle w, x_i \rangle + b\right) \geq 1 - \xi_i \quad \forall 1 \leq i \leq m \\ & \xi_i \geq 0 \end{array} \tag{2}$$

Here, the constraint $y_i\left(\langle w, x_i \rangle + b\right) \geq 1$ ensures that each (x_i, y_i) pair is classified correctly. The slack variable ξ_i relaxes this condition at penalty $C\xi_i$. Finally, minimization of $\|w\|^2$ ensures maximization of the margin by seeking the smallest $\|w\|$ for which the condition $y_i\left(\langle w, x_i \rangle + b\right) \geq 1$ is still satisfied.

2.1 Kernel Expansion

To obtain a nonlinear classifier, one simply replaces the observations x_i by $\Phi(x_i)$. That is, we extract *features* $\Phi(x_i)$ from x_i and compute a linear classifier in terms of the features. Note that there is no need to compute $\Phi(x_i)$ explicitly, since Φ only appears in terms of dot products:

- $\langle \Phi(x), w \rangle$ can be computed by exploiting the linearity of the scalar product, which leads to $\sum_i \alpha_i y_i \langle \Phi(x), \Phi(x_i) \rangle$.

- Likewise $\|w\|^2$ can be expanded in terms of a linear combination scalar products by exploiting the linearity of scalar products twice to obtain $\sum_{i,j} \alpha_i \alpha_j y_i y_j \langle \Phi(x_i), \Phi(x_j) \rangle$.

Furthermore, if we define

$$k(x, x') := \langle \Phi(x), \Phi(x') \rangle, \tag{3}$$

we may use $k(x, x')$ wherever $\langle x, x' \rangle$ occurs. This is often referred to as the *kernel trick* and the resulting hyperplane (now in feature space) is written as

$$f(x) = \langle \Phi(x), w \rangle + b = \sum_{i=1}^{m} \alpha_i y_i k(x_i, x) + b. \tag{4}$$

The family of methods which relies on the kernel trick are popularly called *kernel methods*, and SVMs are one of the most prominent kernel methods.

2.2 Kernels and Microarrays

In evaluation studies comparing different classification techniques, SVMs outperformed all competitors such as Fisher's linear discriminant, Parzen windows and two decision tree learners [6, 13]. Kernel methods have been applied to classify microarray data in numerous studies such as [6], and [9]. An increasing number of application studies in medicine and pharmacogenomics utilize kernel methods to disease, especially cancer diagnosis (for example [8]). Although these studies dealt with time series of microarray measurements, they ignore the temporal character of the data; i.e. microarray data are compared without exploiting the fact that measurement i follows measurement $i - 1$. As these dynamics might reflect, for example, patients' reaction to drug therapy or infection with a pathogen, pharamacogenomics requires a classification method based on the temporal dynamics in gene expression profiles.

A key advantage of kernel methods is that meaningful classifiers can be constructed from non-vectorial data if a sufficiently expressive kernel function can be designed. We are therefore not restricted to vectorial representations when classifying time series of microarray expression data. We exploit this property, in the sequel, by defining a kernel on dynamical systems for time series gene expression profiles.

3 Kernels on Time Series Microarray Data

In this section, we discuss how time series microarray data can be modeled as a Linear Time Invariant (LTI) dynamical system. We then present sub-optimal,

but fast, methods for identifying model parameters. Using these estimated parameters we define kernels on dynamical systems to compare different time series which, in turn, are used by a SVM for classification.

3.1 The Model

We begin by modeling time series microarray data as a partially observed discrete time LTI model. These models are also popular in control theory where they are often called Auto Regressive Moving Average (ARMA) models or Kalman Filters. The time-evolution of this model is described as

$$y_t = Px_t + w_t \text{ where } w_t \sim \mathcal{N}(0, R) \tag{5a}$$
$$x_t = Qx_{t-1} + v_t \text{ where } v_t \sim \mathcal{N}(0, S). \tag{5b}$$

Here, $y_t \in \mathbb{R}^m$ is observed, $x_t \in \mathbb{R}^n$ is the *hidden* or *latent* variable, and $P \in \mathbb{R}^{m\times n}, Q \in \mathbb{R}^{n\times n}, R \in \mathbb{R}^{m\times m}$ and, $S \in \mathbb{R}^{n\times n}$, moreover R and S are positive semi-definite matrices. Typically $m \gg n$, and we set $P^\top P = \mathbf{1}$ to fix the scaling (see e.g. Section 4 [7]). The way to understand these models is to assume that the actual dynamics of the system are guided by a very small number of latent variables while the output space might be very high-dimensional. For instance, gene expression profiles might contain many thousands of genes, while only a few factors may be responsible for the expression.

3.2 Estimating the Parameters

Given a sequence of τ observations the identification problem is to estimate the model parameters P, Q, R and S. Exact solutions like the `n4sid` method in the systems identification toolbox of MATLAB(TM) exist, but they are very expensive to compute when the output space is high dimensional. Instead, we use a sub-optimal closed form solution proposed by [7]. Set $Y := [y_1, \ldots, y_\tau]$, $X := [x_1, \ldots, x_\tau]$, and $W := [w_1, \ldots, w_\tau]$ and solve

$$\min ||Y - PX||_F = \min ||W||_F \text{ such that } P \in \mathbb{R}^{m\times n} \text{ and } P^\top P = \mathbf{1}\,.$$

The unique solution to the above problem is given by $\hat{P} = U_n$ and $\hat{X} = \Sigma_n V_n^\top$ where $U_n \Sigma_n V_n^\top$ is the best rank n approximation of Y. U_n, Σ_n and V_n can be estimated in a straightforward manner from the $Y = U\Sigma V^\top$, the Singular Value Decomposition (SVD) of Y.

In order to estimate Q we solve $\min ||Q\hat{X} - X'||_F$ where $X' = [x_2, \ldots, x_\tau]$ which again has a closed form solution using SVD. Now, we can compute

$\hat{v}_t = x_t - Qx_{t-1}$, and set

$$S = \frac{1}{\tau - 1} \sum_{i=1}^{\tau-1} \hat{v}_i \hat{v}_i^\top .$$

The covariance matrix R can also be computed from the columns of W in a similar manner. For more information including details of efficient implementation we refer the interested reader to [7].

3.3 Dynamical System Kernels

For the sake of defining kernels we use the behavioral framework of [15] and identify dynamical systems, $X := (P, Q, R, S, x_0)$, with their trajectories $\text{Traj}(X) = \{y_t | t \in 1, \ldots, \infty\}$. These trajectories can now be interpreted as linear operators mapping from $\mathbb{R}^m$ (the space of observations y) into the time domain ($\mathbb{N}$ in discrete time systems).

By using a exponentially decaying weighting factor we can define the kernel between two dynamical systems X and X' as the dot product of the trajectories. In other words:

$$k(X, X') := \sum_{t=1}^{\infty} e^{-\lambda t} y_t^\top y'_t = \text{tr}(\text{Traj}(X)\ T\ \text{Traj}(X')^\top), \tag{6}$$

where T is a diagonal operator with entries $e^{-\lambda t}$. Since y_t, y'_t are random variables (5a), we also need to take expectations over w_t, v_t, w'_t, v'_t. Some tedious yet straightforward algebra [14] allows us to compute (6) as follows:

$$k(X, X') = x_0^\top M_1 x'_0 + \frac{1}{e^\lambda - 1} \text{tr}\left[SM_2 + R\right], \tag{7}$$

where x_0 and x'_0 are the initial conditions, and M_1, M_2 satisfy the Sylvester equations:

$$M_1 = e^{-\lambda} Q^\top P^\top P' Q' + e^{-\lambda} Q^\top M_1 Q' \text{ and } M_2 = P^\top P' + e^{-\lambda} Q^\top M_2 Q'. \tag{8}$$

The important point to note is that even though the kernel involves summing over infinite terms, they can be computed in $O(m^3)$ time. These kernels can also be interpreted in a more general framework using the Binet-Cauchy formula. More details can be found in [14].

In our experiments we will use this kernel to compute similarities between gene expression profiles, after having encoded the latter as a dynamical system. This approach has the further advantage that it allows us to compare sequences of different lengths, as they are all mapped to dynamical systems in the first place.

4 Experiments

4.1 Drug response prediction

A central prognosis problem in pharmacogenomics is drug response prediction. We therefore tested our dynamical systems kernel classifier on a set of microarray expression time series data from [4], to predict whether Multiple Sclerosis patients will respond well to treatment with recombinant human interferon beta (rIFNβ).

The dataset contains time expression profiles of 52 multiple sclerosis patients, out of which 33 were good and 19 were poor responders to rIFNβ. Expression profiles of 70 genes were measured for up to seven times per patient; the first five observations were at a regular interval of 3 months each while the last two observations were spaced 6 months apart. 17 patients missed a test and hence have only 6 measurements, 8 patients missed two tests and hence have only 5 measurements.

On this data, Baranzini et al. aimed at determining higher-order predictive patterns associated with treatment outcome and tried to uncover key players (i.e. responsible genes) associated with a good or poor response. When searching for gene expression signatures associated with drug response, they conducted clustering of samples using normalized data for all 70 genes at each time point. Although they applied different similarity measures and clustering algorithms, they did not observe concomitant segregation of samples according to their responder status. They therefore applied a method for feature selection which allowed them to determine triplets of genes whose early expression correlates with responder status. We were interested in the question of predicting user response to the drug using our dynamical systems kernels. As a side effect, we also wanted to estimate the latent dimension of the system, i.e. the number of factors which actually leads to the observed behavior.

For our experiments we modeled the temporal microarray data from [4] as LTI dynamical systems. The parameters of our LTI model were estimated using the methods described in Section 3.2. λ was set to 10 by cross-validation, n was set to a default value of 1. We then computed the kernel matrix for all pairs of dynamical systems and used a SVM for classification. We took 100 random splits of the data into 4 folds of equal size and performed 4-fold cross-validation on each of these splits. We report our classification accuracies as averages over these 100 repetitions.

Our classifier achieved an average prediction accuracy of 87.05% which compares very favorably with the 87.8% accuracy reported by [4]. We must note here that our results are obtained without any specialized knowledge of the dataset or feature selection methods; Baranzini et al. obtained their

result by selecting features, namely triplets of genes whose early expression levels correlate best with response outcome. Furthermore, standard state of the art clustering techniques such as two-way hierarchical clustering failed to satisfactorily separate these samples (see Figure 1 in [4]).

In our second experiment, we checked the influence of the number of measurements per patient on classification accuracy. From a pharmacogenomics point of view, it is interesting to know if we are already able to predict therapy outcome correctly after few measurements. We repeated our classification experiment considering the first k measurements with $k \in \{2, ..7\}$ (at least two measurements are needed to derive a dynamical system). If a patient's data contains less than k measurements, we derive the dynamical systems using all available measurements. As before, in Figure 1 we report results as average classification accuracy of 4-fold cross-validation repeated 100 times.

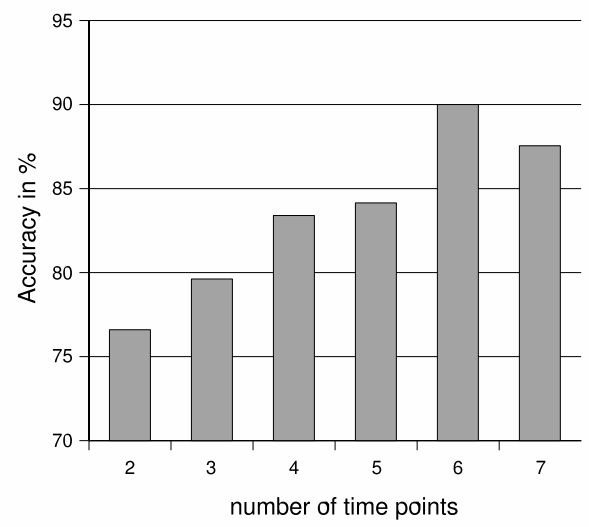

Figure 1: Mean classification accuracy in 100 repetitions versus number of measurements.

As expected, classification accuracy steadily increases as the number of measurements grows. After 3 measurements we already reach a prediction accuracy of more than 80%. But curiously enough, classification accuracy after 7 observations is less than the classification accuracy after 6 observations. This can be explained as follows: The last two observations are taken at a

different time interval (6 months) than the first five observations (3 months). Therefore, the last two observations encode longer range interactions. But, for some patients only six or less observations are available. This means that we are not able to effectively model this long range interaction for those patients and this is reflected in the classification accuracy.

Also observe that by looking at only the first 6 measurements leads to an average classification accuracy of around 90%. This significant jump in classification accuracy (from around 83.5% for 5 measurements) can also be attributed to the modeling of long range effects between 5th and 6th measurement.

Observe that for 6 measurements, we now found a mean accuracy of 89.81% which is significantly better than that of the best-scoring gene triplet reported in [4] with 87.8% accuarcy (Yates' corrected $\chi^2 = 10.26$, $P = 0.0014$).

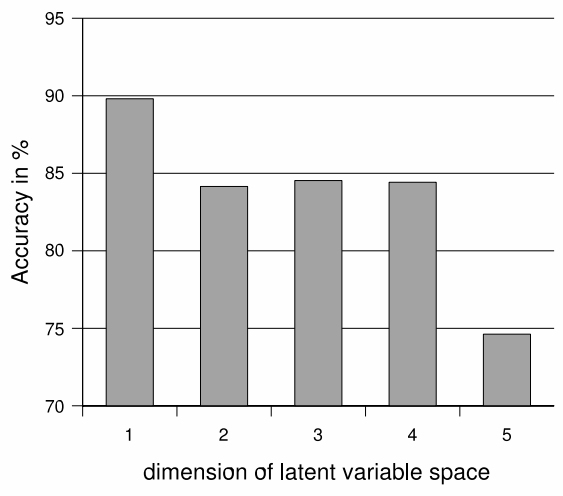

Figure 2: Mean classification accuracy in 100 repetitions versus dimension of latent variable space.

As a final experiment, we tested the impact of the latent dimension n of the system on classification accuracy. All results reported so far were with $n = 1$ which was chosen using cross-validation. We repeated our classification experiment once for each n in $\{2, \ldots, 5\}$. The best accuracy was achieved when we considered only 6 measurements. Results of 4-fold cross-validation repeated

100 times is reported in Figure 2.

Interestingly, a one-dimensional latent variable is the best choice for our dynamical systems model. As a consequence, one single factor, i.e., one group of genes with highly correlated expression levels, seems to be responsible for the microarray time series expression data which allow us to separate good and poor responders with high accuracy.

5 Discussion

In this paper, we modeled a series of gene expression profiles as LTI dynamical systems, and used a kernel on dynamical systems to classify them. The main advantage of our method is that it respects the temporal nature of the data, and is independent of the length of the time series being compared.

Our model can also be extended to predict future values of the time series. In other words, after building the LTI model, we can simulate it to predict the value of a gene expression profile at a future time step. This could have potential applications in disease progression prognosis or in simulations of the cellular gene expression control network.

Ideally, domain knowledge about the underlying dynamics of the data should be used to estimate the latent dimension of our model. But in some cases this knowledge might not be available, and we might have to resort to methods like Locally Linear Embedding (LLE) which estimate the *effective* data dimension.

Our model assumes that the time series has been sampled at constant discrete time intervals. If this assumption is violated then we can use Kalman Filtering to predict the value of the missing observations. In our experiments, the results are encouraging even though we ignore the missing observations. Future work will focus on addressing this issue.

Furthermore, we assume that gene expression level dynamics can be modeled as a linear process. While this appears to be a rather simplistic assumption, our experiments verify that it is rich enough to capture complex dynamics. But, if one examines gene expression dynamics that evolve nonlinearly, our approach might fail to describe the underlying biological process appropriately. In these cases we need to resort to more advanced methods described in [14].

We have shown the potential of our method in drug response prediction in our experiments, but our method offers far more possibilities for applications in pharmacogenomics and medicine, namely in disease diagnosis, and disease and therapy outcome prognosis. For example, one could predict if cancer patients should continue to receive chemotherapy, given their response to initial treatments. Besides, observing microarray time series data, one could try to

predict if a person has been infected by a pathogen or has been exposed to extreme environmental conditions such as heat or stress at a certain point in time.

Furthermore, it will be interesting to combine our classification approach with two central topics of interest in microarray data analysis, namely methods to detect genes that are key players in biological processes and to derive gene regulatory networks from gene expression data.

Acknowledgments

This work was supported in part by National ICT Australia and by the German Ministry for Education, Science, Research and Technology (BMBF) under grant no. 031U112F within the BFAM (Bioinformatics for the Functional Analysis of Mammalian Genomes) project which is part of the German Genome Analysis Network (NGFN). National ICT Australia is funded through the Australian Government's *Backing Australia's Ability* initiative, in part through the Australian Research Council.

References

[1] Bruce Alberts, Alexander Johnson, Julian Lewis, Martin Raff, Keith Roberts, and Peter Walter. *Molecular Biology of the Cell, Fourth Edition*. Garland Science, New York, 2002.

[2] C. A. Ball, I. A. Awad, J. Demeter, J. Gollub, J. M. Hebert, T. Hernandez-Boussard, H. Jin, J. C. Matese, M. Nitzberg, F. Wymore, Z. K. Zachariah, P. O. Brown, and G. Sherlock. The stanford microarray database accommodates additional microarray platforms and data formats. *Nucleic Acids Res*, 33(Database issue):D580–D582, Jan 2005.

[3] Z. Bar-Joseph. Analyzing time series gene expression data. *Bioinformatics*, 20(16):2493–2503, Nov 2004.

[4] S. E. Baranzini, P. Mousavi, J. Rio, S. J. Caillier, A. Stillman, P. Villoslada, M. M. Wyatt, M. Comabella, L. D. Greller, R. Somogyi, X. Montalban, and J. R. Oksenberg. Transcription-based prediction of response to IFNbeta using supervised computational methods. *PLoS Biology*, 3(1):e2, Jan 2005.

[5] T. Barrett, T. O. Suzek, D. B. Troup, S. E. Wilhite, W. C. Ngau, P. Ledoux, D. Rudnev, A. E. Lash, W. Fujibuchi, and R. Edgar. NCBI

GEO: mining millions of expression profiles–database and tools. *Nucleic Acids Res*, 33(Database issue):D562–D566, Jan 2005.

[6] M. P. Brown, W. N. Grundy, D. Lin, N. Cristianini, C. W. Sugnet, T. S. Furey, J. r. Ares M, and D. Haussler. Knowledge-based analysis of microarray gene expression data by using support vector machines. *Proc Natl Acad Sci U S A*, 97(1):262–267, Jan 2000.

[7] G. Doretto, A. Chiuso, Y.N. Wu, and S. Soatto. Dynamic textures. *International Journal of Computer Vision*, 51(2):91–109, 2003.

[8] Y. Lee and C. K. Lee. Classification of multiple cancer types by multicategory support vector machines using gene expression data. *Bioinformatics*, 19(9):1132–1139, Jun 2003.

[9] G. Natsoulis, L. El Ghaoui, G. R. Lanckriet, A. M. Tolley, F. Leroy, S. Dunlea, B. P. Eynon, C. I. Pearson, S. Tugendreich, and K. Jarnagin. Classification of a large microarray data set: algorithm comparison and analysis of drug signatures. *Genome Research*, 15(5):724–736, May 2005.

[10] M. F. Ramoni, P. Sebastiani, and I. S. Kohane. Cluster analysis of gene expression dynamics. *Proceedings of the National Academy of Science*, 99(14):9121–9126, Jul 2002.

[11] B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

[12] P. T. Spellman, G. Sherlock, M. Q. Zhang, V. R. Iyer, K. Anders, M. B. Eisen, P. O. Brown, D. Botstein, and B. Futcher. Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. *Mol Biol Cell*, 9(12):3273–3297, Dec 1998.

[13] A. Statnikov, C. F. Aliferis, I. Tsamardinos, D. Hardin, and S. Levy. A comprehensive evaluation of multicategory classification methods for microarray gene expression cancer diagnosis. *Bioinformatics*, 21(5):631–643, Mar 2005.

[14] S. V. N. Vishwanathan, R. Vidal, and A. J. Smola. Kernels and dynamical systems. *Automatica*, 2005. submitted.

[15] J. C. Willems. From time series to linear system. I. Finite-dimensional linear time invariant systems. *Automatica J. IFAC*, 22(5):561–580, 1986.