Efficient Sparse Modeling with Automatic Feature Grouping

Leon Wenliang Zhong WZHONG@CSE.UST.HK
James T. Kwok JAMESK@CSE.UST.HK
Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong

Abstract

The grouping of features is highly beneficial in learning with high-dimensional data. It reduces the variance in the estimation and improves the stability of feature selection, leading to improved generalization. Moreover, it can also help in data understanding and interpretation. OSCAR is a recent sparse modeling tool that achieves this by using a ℓ_1-regularizer and a pairwise ℓ_∞-regularizer. However, its optimization is computationally expensive. In this paper, we propose an efficient solver based on the accelerated gradient methods. We show that its key projection step can be solved by a simple iterative group merging algorithm. It is highly efficient and reduces the empirical time complexity from $O(d^3 \sim d^5)$ for the existing solvers to just $O(d)$, where d is the number of features. Experimental results on toy and real-world data sets demonstrate that OSCAR is a competitive sparse modeling approach with the added ability of automatic feature grouping.

1. Introduction

Many real-world data sets are high-dimensional and contain spurious features. Sparse modeling, which selects a relevant subset of features while learning the model, is thus often indispensable. It has been used in diverse application areas such as computer vision, signal processing, and bioinformatics.

Lasso (Tibshirani, 1996) is the most popular sparse modeling algorithm. However, in the presence of highly correlated features, it tends to select only one of them. Consequently, estimation can be unstable and the resultant model is difficult to interpret.

Appearing in *Proceedings of the 28th International Conference on Machine Learning*, Bellevue, WA, USA, 2011.

Another deficiency with lasso is that it cannot find feature groups. In general, feature grouping can reduce the variance of the estimator (Shen & Huang, 2010) and improve the stability of feature selection (Jörnsten & Yu, 2003). It also helps to gain additional insight on the underlying data generation process, such as in the finding of co-regulated genes in microarray analysis (Dettling & Bühlmann, 2004). Note that this is different from the group lasso (Yuan & Lin, 2006), which requires the feature groups to be known in advance.

A simple approach to find feature groups is by first performing unsupervised clustering on the data, and then feed the cluster centers as new features to a supervised learning method (Hastie et al., 2001). A more desirable approach is to integrate these two steps. Dettling & Bühlmann (2004) used a heuristic strategy that grows and prunes the feature groups incrementally. Jörnsten & Yu (2003) combined feature clustering and classification into a single MDL code, and then use a search procedure to find the best model.

The explicit search for feature groups is a combinatorial optimization problem. Instead, one can use an appropriate regularizer to encourage its formation. A well-known approach is the elastic net (Zou & Hastie, 2005). In situations where the features are ordered in some meaningful way, the fused lasso (Tibshirani et al., 2005) directly encourages the successive feature coefficients to be similar. However, oftentimes such an ordering does not exist naturally, and needs to be estimated before the fused lasso can be used.

Recently, two approaches, namely the grouping pursuit (Shen & Huang, 2010) and OSCAR (Octagonal Shrinkage and Clustering Algorithm for Regression) (Bondell & Reich, 2008), are proposed for the more challenging feature grouping problem when features are not ordered. Similar to the fused lasso, they try to pull two feature coefficients β_i and β_j together. Specifically, grouping pursuit uses the regularizer $\sum_{j<j'} G(\beta_j - \beta_{j'})$, where $G(z) = \lambda$ (a regularization parameter) if $|z| > \lambda$; and $|z|$ otherwise. However, since $G(\cdot)$ is nonconvex, this leads to a sequence

of computationally expensive difference convex (DC) programs. Moreover, it cannot obtain sparse solutions.

In this paper, we will focus on the OSCAR. With a novel pairwise ℓ_∞ norm, it encourages both sparsity and equality of coefficients for correlated features. Feature groups are automatically discovered simultaneously with regression shrinkage, without the need to pre-specify the grouping structures as in group lasso. Moreover, since the coefficients for the grouped features are tied, the resultant model has a reduced model complexity and is less prone to over-fitting.

Despite these advantages, the optimization problem of OSCAR, though still convex, is much more challenging. Bondell and Reich (2008) proposed two solvers. The first one involves a huge quadratic program (QP) with $O(d^2)$ variables and $O(d^2)$ constraints, where d is the number of features. The second approach involves a sequence of (potentially smaller) QP's with an increasing number of constraints, which however can go up to $O(d!)$ in the worst case. Hence, both solvers are not scalable and the experiments in (Bondell & Reich, 2008) are limited to small feature sets. Alternatively, as OSCAR's pairwise ℓ_∞ norm is simply the sum of ℓ_∞ norms over groups of two variables, one can use the network flow algorithm recently proposed in (Mairal et al., 2010a). However, this algorithm is designed for general overlapping groups but not tailored for OSCAR. Because of the $O(d^2)$ number of groups in OSCAR, each iteration will involve solving a maxflow problem on a canonical graph $G = (V, E)$ with $|V| = |E| = O(d^2)$, and results in a complexity of $O(|V|^2|E|^{\frac{1}{2}}) = O(d^5)$ (Mairal et al., 2010b).

In this paper, we propose an accelerated gradient algorithm (Beck & Teboulle, 2009) that is tailored for OSCAR's optimization problem. By using a simple group merging algorithm, the key projection step can be solved exactly and efficiently in $O(d\log(d))$ time. Hence, the proposed algorithm is particularly efficient on high-dimensional data sets.

The rest of this paper is organized as follows. In Section 2, we first give a brief review on OSCAR and accelerated gradient methods. Section 3 then describes the proposed solver. Experimental results are presented in Section 4, and the last section gives some concluding remarks.

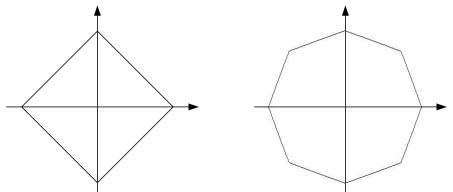
2. Related Work

2.1. OSCAR

Let $\mathbf{X} \in \mathbb{R}^{n\times d}$ be the input data matrix (with each row being an instance) and $\mathbf{y} \in \mathbb{R}^n$ be the corresponding output. We assume that $\mathbf{y}$ is centered, and each column of $\mathbf{X}$ is standardized. OSCAR is formulated as the following optimization problem:

$$\min_{\boldsymbol{\beta}} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda_1\|\boldsymbol{\beta}\|_1 + \lambda_2 \sum_{i<j} \max\{|\beta_i|, |\beta_j|\}, \quad (1)$$

where λ_1, λ_2 are regularization parameters. The OSCAR regularizer consists of two parts: An ℓ_1-regularizer which encourages sparsity as in lasso, and a pairwise ℓ_∞-regularizer which encourages every coefficient pairs $|\beta_i|, |\beta_j|$ to be equal (Figure 1).



(a) Lasso. (b) OSCAR.

Figure 1. Constraint regions for lasso and OSCAR.

Bondell and Reich (2008) proposed two solvers for (1). Unfortunately, both of them are computationally expensive. The first solver involves a huge QP with $(d^2 + 3d)/2$ variables and $d^2 + d + 1$ linear constraints. The second one (described in the web appendix B of (Bondell & Reich, 2008)) uses a sequential QP algorithm in which constraints are gradually added, but the total number of constraints can be $O(d!)$.

2.2. Accelerated Gradient Methods

Gradient methods are well-known for their simplicity and scalability. However, a major drawback is that they have slow convergence. In the past decades, attempts were made to accelerate gradient methods. Nesterov (1983) pioneered the "optimal method" for smooth optimization, which achieves the optimal convergence rate for a black-box model. Subsequent works (Beck & Teboulle, 2009; Nesterov, 2007) extended this to composite optimization problems of the form $\min_{\boldsymbol{\beta}} \; f(\boldsymbol{\beta}) + r(\boldsymbol{\beta})$, where $f(\boldsymbol{\beta})$ is convex with L-Lipschitz continuous gradient, and $r(\boldsymbol{\beta})$ is convex but nonsmooth. While gradient methods perform descent by simply using the (sub)gradient, accelerated methods first solve the following optimization problem (often called the *projection* or *proximal* step)

$$\begin{aligned}\arg\min_{\boldsymbol{\beta}} Q(\boldsymbol{\beta}; \hat{\boldsymbol{\beta}}^t) \quad &\equiv \quad (\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}^t)^T \nabla f(\hat{\boldsymbol{\beta}}^t) \\ &\quad + \frac{L}{2}\|\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}^t\|^2 + r(\boldsymbol{\beta}), \quad (2)\end{aligned}$$

where $\hat{\boldsymbol{\beta}}^t$ is the current estimate at iteration t. Note that $Q(\boldsymbol{\beta}; \hat{\boldsymbol{\beta}}^t)$ is a quadratic approximation on the

smooth component $f(\boldsymbol{\beta})$, while leaving the nonsmooth component $r(\boldsymbol{\beta})$ intact. Problem (2) is also a common construct in the proximal methods (Bach et al., 2011) and many recent stochastic subgradient methods (Duchi & Singer, 2009).

The subsequent update step depends on the specific accelerated algorithm. In this paper, we will adopt the FISTA (Fast Iterative Shrinkage-Thresholding Algorithm) (Beck & Teboulle, 2009), which is shown in Algorithm 1. While traditional gradient methods have a slow convergence rate of $O(1/\sqrt{N})$, where N is the number of iterations, FISTA converges as $O(1/N^2)$. However, this requires that the crucial projection step in (2) can be solved efficiently and exactly.

Algorithm 1 FISTA (Beck & Teboulle, 2009).

1: **Initialize:** $\hat{\boldsymbol{\beta}}^1 \leftarrow \boldsymbol{\beta}^0 \in \mathbb{R}^n,\ \tau_1 \leftarrow 1$.
2: **for** $t = 1, 2, \ldots, N$ **do**
3: $\quad \boldsymbol{\beta}^t \leftarrow \arg\min_{\boldsymbol{\beta}}\ Q(\boldsymbol{\beta}; \hat{\boldsymbol{\beta}}^t)$. {projection step}
4: $\quad \tau_{t+1} \leftarrow \frac{1+\sqrt{1+4\tau_t^2}}{2}$.
5: $\quad \hat{\boldsymbol{\beta}}^{t+1} \leftarrow \boldsymbol{\beta}^t + \left(\frac{\tau_t - 1}{\tau_{t+1}}\right)(\boldsymbol{\beta}^t - \boldsymbol{\beta}^{t-1})$.
6: **end for**
7: Output $\boldsymbol{\beta}^N$.

2.3. Structured Sparse Models

The OSCAR regularizer is an example of structured sparsity-inducing norm (Bach et al., 2011; Jenatton et al., 2009). Recently, Mairal *et. al.* (2010a) considered structured norms of the form

$$\sum_{g \in \mathcal{G}} \eta_g \|\boldsymbol{\beta}_g\|_\infty, \tag{3}$$

where $\mathcal{G}$ is an arbitrary set of overlapping groups of indices in $\{1, 2, \ldots, d\}$, $\boldsymbol{\beta}_g$ is the subvector of $\boldsymbol{\beta}$ indexed by group g, and η_g the corresponding weight. They developed an efficient algorithm (called ProxFlow) which is based on accelerated gradient methods. As discussed in Section 2.2, its success thus relies on the efficient computation of the projection step. It is shown that the dual of this step can be reformulated as a quadratic min-cost flow algorithm on a canonical graph $G(V, E)$, where $|V| = |\mathcal{G}| + d$ and $|E| = |\mathcal{G}| + \sum_{g \in \mathcal{G}} |g| + d$. The worst-case complexity for solving this maxflow problem is $O(|V|^2 |E|^{\frac{1}{2}})$ (Mairal et al., 2010b), though empirically it can be much faster.

Obviously, (3) admits the OSCAR regularizer in (1) as a special case, and thus ProxFlow can be readily used for its optimization. However, because of the $O(d^2)$ number of groups in OSCAR's pairwise ℓ_∞-regularizer, the canonical graph in ProxFlow's maxflow problem has $|V| = |E| = O(d^2)$. Consequently, each projection step takes $O(d^5)$ time. As will be shown in Section 4, empirically this is as slow as the solvers in (Bondell & Reich, 2008).

Recently, Bach (2010) proposed an interesting connection between structured sparsity-inducing norms (including that of OSCAR) and submodular functions. The projection step can then be cast as submodular function minimization, in which standard algorithms can be used. However, these algorithms either have high complexity (e.g., $O(d^6)$) or no complexity bound at all (e.g., the minimum-norm-point algorithm) (Fujishige, 2005). Moreover, as these solvers are generic, they can be rather inefficient for a specific model.

3. Efficient Projection Step for OSCAR

In this section, we will show that an efficient algorithm can be tailor-made for the projection step of OSCAR. First, we decompose the OSCAR objective in (1) as

$$\begin{aligned} f(\boldsymbol{\beta}) &= \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2, \\ r(\boldsymbol{\beta}) &= \lambda_1 \|\boldsymbol{\beta}\|_1 + \lambda_2 \sum_{i<j} \max\{|\beta_i|, |\beta_j|\}, \end{aligned}$$

where $f(\boldsymbol{\beta})$ is smooth and $r(\boldsymbol{\beta})$ is nonsmooth. The objective $Q(\boldsymbol{\beta}; \hat{\boldsymbol{\beta}}^t)$ in (2) can then be written as

$$\min_{\boldsymbol{\beta}} \sum_{i=1}^{d} \beta_i^2 - \frac{2[(L\hat{\beta}_i^t - \nabla f_i^t)\beta_i]}{L} + \frac{2}{L} r(\boldsymbol{\beta}), \tag{4}$$

where $f_i^t = [f(\hat{\boldsymbol{\beta}}^t)]_i$. Note that the only term that depends on the sign of β_i is $(L\hat{\beta}_i^t - \nabla f_i^t)\beta_i$. To minimize (4), one has to choose the sign of β_i such that

$$(L\hat{\beta}_i^t - \nabla f_i^t)\beta_i \geq 0. \tag{5}$$

Thus (4) can be rewritten more compactly as

$$\min_{\mathbf{z} \geq \mathbf{0}}\ F(\mathbf{z}) \equiv (\mathbf{z} - \mathbf{a})^T \mathbf{z} + \hat{r}(\mathbf{z}), \tag{6}$$

where

$$z_i = |\beta_i|,\ \ a_i = 2\frac{|L\hat{\beta}_i^t - \nabla f_i^t|}{L},\ \ \hat{r}(\mathbf{z}) = \frac{2}{L} r(\mathbf{z}). \tag{7}$$

Next, we permute a_i's (and z_i's accordingly) such that

$$a_1 \geq \cdots \geq a_d \geq 0. \tag{8}$$

3.1. Properties of the Optimal Solution of (6)

3.1.1. z_i^*'s are Non-increasing

With the a_i's sorted in decreasing order as in (8), the following proposition shows that the optimal z_i^*'s will also be arranged in the same order. (Proofs are omitted due to lack of space.)

Proposition 1. *For the optimal solution in (6),*

$$z_1^* \geq z_2^* \geq \ldots \geq z_d^* \geq 0. \tag{9}$$

Moreover, if $a_i = a_j$, then $z_i^* = z_j^*$

By sorting the $|\beta_i|$'s in (1) to form $|\beta_{(i)}|$'s such that $|\beta_{(1)}| \geq |\beta_{(2)}| \geq \cdots \geq |\beta_{(d)}|$, the OCSAR regularizer can be written as $\lambda_1\|\boldsymbol{\beta}\|_1 + \lambda_2 \sum_{i=1}^d (d-i)|\beta_{(i)}|$. In other words, z_i^*, the ith largest value among the z_i^*s, has a weight of $w_i = \frac{2}{L}[\lambda_1 + \lambda_2(d-i)]$ associated with z_i^*. We can then replace the constraint $\mathbf{z} \geq \mathbf{0}$ in (6) by the more explicit ordering constraint in (9):

$$\begin{aligned} \min_{\mathbf{z}} \quad & F(\mathbf{z}) = \sum_{i=1}^d \bar{F}_i(z_i) \\ \text{s.t.} \quad & z_1 \geq z_2 \geq \cdots \geq z_d \geq 0, \end{aligned} \tag{10}$$

where $\bar{F}_i(z_i) = (z_i - a_i)z_i + w_i z_i$.

3.1.2. Groups in $\mathbf{z}^*$

Since the OSCAR regularizer encourages $|\beta_i| = |\beta_j|$, we expect that some of $\{z_1, \ldots, z_d\}$ will be lumped into groups. The following gives a formal definition.

Definition 1. *Given $\{z_1, \ldots, z_d\}$, the set of indices $I_{s:t} = \{i \in \mathbb{Z} : s \leq i \leq t\}$, where $s, t \in \mathbb{Z}$, is called a* group *(denoted $\mathcal{G}_{s:t}$) if*

1. *$z_i = z_j$ for all $i, j \in I_{s:t}$;*
2. *$z_{s-1} \neq z_s$ if $s \neq 1$;*
3. *$z_t \neq z_{t+1}$ if $t \neq d$.*

If we consider a particular group $\mathcal{G}_{s:t}$ in isolation, the corresponding minimum value of the sub-sum $\sum_{i \in \mathcal{G}_{s:t}} \bar{F}_i(z_i)$ can be obtained from the following lemma.

Lemma 1. *Given a group $\mathcal{G}_{s:t}$, $\sum_{i \in \mathcal{G}_{s:t}} \bar{F}_i(z_i)$ reduces to the univariate convex quadratic function $\sum_{i \in \mathcal{G}_{s:t}} \bar{F}_i(z)$, which is minimized by the* group value

$$z = z_s = \ldots = z_t = \max\{v_{s:t}, 0\}, \tag{11}$$

where $v_{s:t} \equiv \frac{\sum_{i \in \mathcal{G}_{s:t}} (a_i - w_i)}{2(t-s+1)}$.

Obviously, at the optimal solution $\mathbf{z}^*$, $\mathcal{G}_{s:t}$ cannot be considered in isolation but needs to be determined together with the other z_i's so that Proposition 1 is satisfied. Interestingly, the following proposition shows that the common value for each $z_i \in \mathcal{G}_{s:t}$ is still given by the group value in (11).

Proposition 2. *For a group $\mathcal{G}_{s:t}$ in the optimal solution $\mathbf{z}^*$ of (10), the common value for each entry in $\mathcal{G}_{s:t}$ is given by the group value in (11).*

Moreover, we will show in Proposition 3 that the groups in the optimal $\mathbf{z}^*$ must be *coherent* in the following sense.

Definition 2. *A group $\mathcal{G}_{s:t}$ is* coherent *if there is no integer $u \in \{s, s+1, \ldots, t-1\}$ such that $v_{s:u} > v_{u+1:t}$.*

Proposition 3. *For any group $\mathcal{G}_{s:t}$ in the optimal $\mathbf{z}^*$, if its group value $v_{s:t} \geq 0$, then $\mathcal{G}_{s:t}$ is coherent.*

Now we are ready to give the sufficient and necessary conditions for the optimal $\mathbf{z}^*$.

Proposition 4. *$\mathbf{z}^*$ is an optimal solution of (10) if and only if it satisfies Propositions 1, 2 and 3.*

3.2. The Algorithm

Initially, every index $s \in \{1, 2, \ldots, d\}$ forms a group $\mathcal{G}_{s:s} = \{s\}$ of its own. The following proposition shows that if we have two consecutive coherent groups $\mathcal{G}_{s:u}$ and $\mathcal{G}_{u+1,t}$ such that the group values are not in decreasing order, then the merged group is also coherent.

Proposition 5. *For two consecutive coherent groups $\mathcal{G}_{s:u}$ and $\mathcal{G}_{u+1,t}$. If $v_{s:u} \leq v_{u+1:t}$, the merged group $\mathcal{G}_{s:t}$ is also coherent.*

Algorithm 2 shows how to obtain the optimal $\mathbf{z}^*$. Initially, each index forms a group. Starting from the first group, it examines the group values of two consecutive groups. If they are not in decreasing order, the groups are merged and pushed to the stack. Hence, when the algorithm terminates, the groups are arranged in decreasing order, and thus satisfies Proposition 1. After obtaining the group structure, the common value for entries in each group of $\mathbf{z}^*$ is simply the group value given by (11), and thus Proposition 2 is also satisfied. Moreover, Proposition 5 ensures that the merged group is coherent, thus satisfying Proposition 3.

Algorithm 2 Solving the projection step.

```
Compute a in (7).
Sort a_i's in decreasing order. Use the sorted indices to form groups G_{1:1}, G_{2:2}, ..., G_{d:d}.
Set stack S = {G_{1:1}} and let G_top be the group at the top of S.
for i = 2, 3, ..., d do
    Ḡ ← G_{i:i}.
    while S is not empty and v(Ḡ) ≥ v(G_top) do
        Ḡ ← Ḡ ∪ G_top. {merge}
        pop G_top from S.
    end while
    push Ḡ onto S.
end for
```

The optimal $\boldsymbol{\beta}^*$ for the projection step can be obtained from $\mathbf{z}^*$ as follows. Its magnitude is recovered from (7)

as $|\beta_i^*| = z_i^*$, while its sign is chosen such that (5) is satisfied.

3.2.1. Time Complexity

Initially, there are d groups and each merge operation reduces the number of groups by one. Hence, there are at most $d-1$ merge operations. As each merge operation and other stack operations take $O(1)$ time, the complexity of Algorithm 2 is dominated by the initial sorting of a_i's, which takes $O(d \log d)$ time. This is much faster than the QP-based and SQP-based solvers discussed in Section 2.1, and the network flow algorithms for general structured sparse models discussed in Section 2.3.

The number of iterations for FISTA to obtain an ϵ-optimal solution is $O(\frac{1}{\sqrt{\epsilon}})$ (Beck & Teboulle, 2009). The time to compute the gradient of $\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2$ is usually $O(nd)$. Hence, the total time is $O\left(\frac{1}{\sqrt{\epsilon}}(d(n + \log d)\right)$. Typically, $n \gg \log d$, and the time thus scales linearly w.r.t. d.

3.2.2. Remarks

As discussed in Section 2.2, the projection step is a core component in many other proximal and accelerated gradient methods. Hence, the proposed efficient computation of the projection step can also be used with these methods and is not limited to FISTA.

Moreover, since an (accelerated) gradient method only requires the knowledge of the gradient, the proposed algorithm can be used to extend OSCAR to other loss functions (besides the square loss). Indeed, even when the loss is non-smooth (such as the hinge loss), one can still utilize the proposed algorithm with methods like FOBOS (Duchi & Singer, 2009) in both the deterministic and stochastic settings.

4. Experiments

In this section, we perform experiments on a number of synthetic and real-world data sets. At discussed in (Shen & Huang, 2010), encouraging coefficients to be similar to each other will unwantedly over-penalize large pairwise coefficient differences and thus impedes performance. Instead of resorting to the use of a non-convex regularizer as in (Shen & Huang, 2010), we will alleviate this problem in OSCAR by taking a two-step approach. First, we run OSCAR to obtain the group structure. Features with zero coefficients are discarded, while those in the same group $\mathcal{G}$ are merged to form a new feature $\mathbf{x}_\mathcal{G} = \sum_{i \in \mathcal{G}} \text{sgn}(\beta_i)\mathbf{x}_i$ of weight $|\mathcal{G}|$. They are then used to train a weighted ridge regression model. The rescaled version will be denoted R-OSCAR. A similar two-stage approach is also implemented for lasso and elastic net. However, they are not as effective and so will not be reported here.

4.1. Efficiency of the Proposed Solver

We compare the proposed solver with (1) the QP and sequential QP (SQP) algorithms[1] in (Bondell & Reich, 2008); and (2) ProxFlow[2] (Mairal et al., 2010b). Following (Bondell & Reich, 2008), the data is generated from the model $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \epsilon$, with $\epsilon \sim N(0, \sigma^2)$ and $\sigma = 15$. We use the two most difficult synthetic problems (examples 4 and 5 in their Section 4), with varying input dimensionality d:

1. $\boldsymbol{\beta} = [\underbrace{0, \ldots, 0}_{0.3d}, \underbrace{2, \ldots, 2}_{0.2d}, \underbrace{0, \ldots, 0}_{0.3d}, \underbrace{2, \ldots, 2}_{0.2d}]^T$, and $\mathbf{X} \sim \mathcal{N}(\mathbf{0}, \mathbf{C})$, where the covariance matrix $\mathbf{C}$ is given by $c_{ij} = 0.5$ for $i \neq j$ and $c_{ii} = 1$.
2. $\boldsymbol{\beta} = [\underbrace{3 \ldots 3}_{0.3d}, \underbrace{0 \ldots 0}_{0.7d}]^T$, and inputs are generated as
$$\begin{aligned} &\mathbf{x}_i = Z_1 + \epsilon_i,\ Z_1 \sim \mathcal{N}(0,1) && i = 1, \ldots, 0.1d, \\ &\mathbf{x}_i = Z_2 + \epsilon_i,\ Z_2 \sim \mathcal{N}(0,1) && i = 0.1d+1, \ldots, 0.2d, \\ &\mathbf{x}_i = Z_3 + \epsilon_i,\ Z_3 \sim \mathcal{N}(0,1) && i = 0.2d+1, \ldots, 0.3d, \\ &\mathbf{x}_i \sim \mathcal{N}(0,1) && i = 0.3d+1, \ldots, d, \end{aligned}$$
and $\epsilon_i \sim \mathcal{N}(0, 0.16)$.

We use 1000 samples for training. Experiments are performed on a PC with a quad-core AMD 3.0GHz CPU and 8GB memory.

On all the runs, the four solvers return almost identical solutions[3], and so only the time is reported in Figure 2. As expected, the proposed solver is much faster than the others when d is large (e.g., $d > 32$). Empirically, the running time of the proposed solver scales as $O(d)$ (which agrees with our analysis in Section 3.2.1), while the QP-based solver scales as $O(d^{4.6})$, SQP scales as $O(d^{3.8})$, and ProxFlow as $O(d^{3.2})$.

4.2. Comparison with Other Sparse Methods

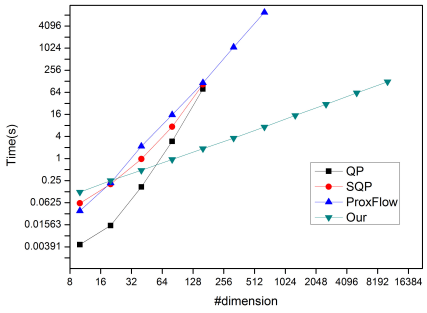
4.2.1. Synthetic Data

We perform experiments on five synthetic problems that have been commonly used in the sparse learning literature (Bondell & Reich, 2008; Tibshirani, 1996; Zou & Hastie, 2005). The regression model is $\mathbf{y} = \mathbf{X}\boldsymbol{\beta}^* + \epsilon$, with $\epsilon \sim N(0, \sigma^2)$ and

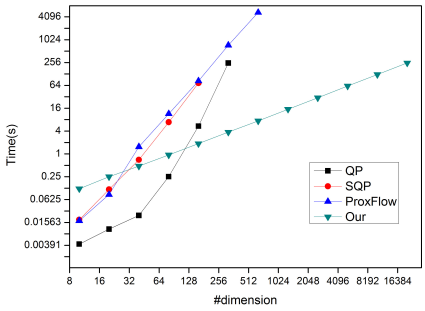
[1] `http://www4.stat.ncsu.edu/~bondell/software.html`

[2] `http://www.di.ens.fr/~mairal/software.php`

[3] The relative differences in the obtained objective values and $\|\boldsymbol{\beta}\|$ are typically smaller than 10^{-5}.



(a) Data set 1.



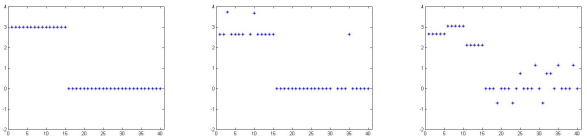
(b) Data set 2.

Figure 2. Running time for various OSCAR solvers. Note that both axes are in log scale.

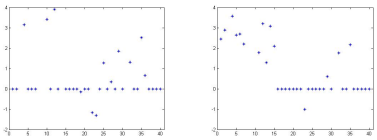
1. $d = 8, \boldsymbol{\beta}^* = [3, 2, 1.5, 0, 0, 0, 0, 0]^T, \sigma = 3$, and $cov(\mathbf{x}_i, \mathbf{x}_j) = 0.7^{|i-j|}$;
2. Same as data set 1 except that $\boldsymbol{\beta}^* = [3, 0, 0, 1.5, 0, 0, 0, 2]^T$;
3. Same as data set 1 except that $\boldsymbol{\beta}^* = [0.85, 0.85, \ldots, 0.85]^T$;
4. Same as data set 1 in Section 4.1, with $d = 40$.
5. Same as data set 2 in Section 4.1, except that $d = 40$ and the groups of nonzero and zero coefficients have sizes 15 and 25, respectively.

Figure 3 shows the feature coefficients obtained by the various methods in a typical run. For lack of space, only results for data set 5 are shown. Feature ordering for the fused lasso is obtained by hierarchical clustering, as suggested in (Tibshirani et al., 2005). Parameters in all the models are tuned using an independent validation set. As can be seen, only R-OSCAR and fused lasso can perform feature grouping, and the R-OSCAR solution is closer to the ground truth.

Table 1 compares the various methods in terms of the (1) mean-squared error (MSE) $(\boldsymbol{\beta} - \boldsymbol{\beta}^*)\mathbf{X}^T\mathbf{X}(\boldsymbol{\beta} - \boldsymbol{\beta}^*)$; (2) degree of freedom (dof), which is the number of unique nonzero coefficients obtained. For OSCAR and R-OSCAR, it is the number of groups discovered (Bondell & Reich, 2008); (3) number of nonzero coefficients that are estimated as zero ($\#(NZ \to Z)$); and (4) number of zero coefficients that are estimated as nonzero ($\#(Z \to NZ)$). As can be seen, R-OSCAR



(a) ground truth. (b) R-OSCAR. (c) fused lasso.



(d) lasso. (e) elastic net.

Figure 3. Feature coefficients obtained on data set 5.

reduces the bias associated with OSCAR and has better MSE. Overall, R-OSCAR outperforms the others on all four criteria. Because of its ability to automatically group features, R-OSCAR reduces the dof and thus model complexity, making it less prone to overfitting.

4.2.2. 20-Newsgroups

In this experiment, we use 5 pairs of newsgroups from the 20-newsgroups data set. The first 3 pairs are neighboring newsgroups in the hierarchy, while the last 2 pairs are distant newsgroups. We remove words that appear in fewer than 3 documents. 20% and 40% of samples are then used for training and validation, respectively, and the rest are for testing. All features are centered and scaled to unit variance. To make the data set more challenging, we first run ridge regression, and then duplicate the 30 most important features 100 times (with added noise generated from $\mathcal{N}(0, 0.16)$) to form 30 additional groups.

Results are shown in Table 2. As can be seen, while the elastic net has the best accuracy, its dof is much larger. In contrary, lasso and fused lasso often have small dofs, but their accuracies are inferior. On the other hand, the accuracy of R-OSCAR is comparable with the elastic net, and yet enjoys a small dof. Figure 4 compares the coefficients obtained by the elastic net and R-OSCAR on the 30 additional groups of the "motorcycles vs. autos" task. Though the elastic net has some grouping effect, it fails to tie the features in the same group (which are generated as duplicates of each other) together. In contrast, the R-OSCAR solution shows much clearer feature grouping.

4.2.3. Breast Cancer

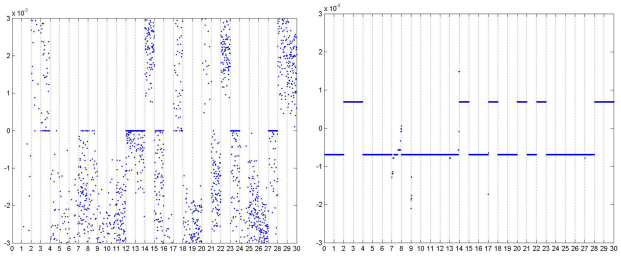
The finding of co-regulated gene groups is very useful in bioinformatics. Here, we experiment with the breast

Table 1. Results on the synthetic data. The numbers in brackets are the standard errors (of the median) estimated by using the bootstrap with $B = 500$ as in (Zou & Hastie, 2005).

		lasso	fused lasso	elastic net	OSCAR	R-OSCAR
data set 1	MSE	2.59(0.53)	1.60(0.29)	**0.84(0.24)**	2.50(0.51)	1.07(0.36)
#train=20	dof	4.87(0.32)	3.42(0.47)	4.20(0.37)	4.93(0.24)	**2.98(0.12)**
#test=200	$\#(Z \to NZ)$	2.09(0.40)	2.29(0.72)	1.28(0.43)	2.84(0.46)	**0.02(0.13)**
	$\#(NZ \to Z)$	**0.00(0.00)**	**0.00(0.00)**	**0.00(0.00)**	**0.00(0.00)**	**0.00(0.00)**
data set 2	MSE	2.31(0.32)	1.96(0.18)	1.58(0.23)	1.94(0.23)	**1.50(0.29)**
#train=20	dof	5.00(0.38)	4.27(0.42)	5.42(0.47)	4.83(0.36)	**3.72(0.47)**
#test=200	$\#(Z \to NZ)$	**2.91(0.28)**	3.52(0.48)	3.20(0.38)	3.93(0.40)	3.39(0.45)
	$\#(NZ \to Z)$	0.98(0.10)	0.96(0.17)	1.00(0.00)	**0.49(0.47)**	0.97(0.17)
data set 3	MSE	2.87(0.43)	0.43(0.247)	1.45(0.21)	1.23(0.40)	**0.08(0.11)**
#train=20	dof	6.01(0.21)	1.02(0.12)	7.38(0.46)	3.47(0.47)	**1.00(0.00)**
#test=200	$\#(Z \to NZ)$	**0.00(0.00)**	**0.00(0.00)**	**0.00(0.00)**	**0.00(0.00)**	**0.00(0.00)**
	$\#(NZ \to Z)$	1.99(0.20)	**0.00(0.00)**	0.59(0.47)	**0.00(0.00)**	**0.00(0.00)**
data set 4	MSE	47.30(2.71)	24.49(1.42)	34.61(1.60)	24.44(1.23)	**20.48(0.71)**
#train=100	dof	19.66(0.60)	**2.75(0.48)**	23.93(0.38)	10.28(1.21)	15.32(3.57)
#test=400	$\#(Z \to NZ)$	**7.00(0.62)**	19.68(0.57)	9.15(0.39)	19.97(0.14)	19.27(0.42)
	$\#(NZ \to Z)$	7.03(0.63)	0.00(0.04)	4.90(0.30)	**0.00(0.00)**	0.00(0.03)
data set 5	MSE	61.71(4.67)	36.38(5.78)	33.25(3.09)	50.48(3.50)	**16.02(1.95)**
#train=50	dof	13.73(0.76)	8.63(0.55)	16.72(0.68)	16.92(0.86)	**5.03(0.26)**
#test=400	$\#(Z \to NZ)$	4.87(0.39)	5.44(0.58)	3.49(0.69)	10.80(1.21)	**0.53(0.47)**
	$\#(NZ \to Z)$	6.39(0.47)	**0.01(0.07)**	1.62(0.46)	0.63(0.52)	0.40(0.46)

Table 2. Results on the 20-newsgroups subset. Numbers in brackets under each data set name are the dimensionality, number of training samples, number of validation samples, and number of test samples.

		lasso	fused lasso	elastic net	OSCAR	R-OSCAR
baseball vs. hockey	test accuracy	84.71	85.46	**87.09**	84.46	86.22
(7579/396/794/798)	dof	234	278	1807	**206**	381
guns vs. mideast	test accuracy	92.09	91.04	**93.02**	91.96	92.09
(9247/364/726/759)	dof	**117**	206	1738	178	238
autos vs. motorcycles	test accuracy	83.04	84.30	**88.19**	84.67	86.06
(8573/394/790/796)	dof	**104**	360	5637	308	246
windows.x vs. misc	test accuracy	86.23	87.40	**89.61**	86.11	88.68
(7351/250/502/857)	dof	500	**213**	3013	373	304
atheism vs. graphics	test accuracy	93.70	93.70	**94.81**	93.83	94.57
(7195/320/638/810)	dof	486	**273**	1872	451	293



(a) elastic net. (b) R-OSCAR.

Figure 4. Coefficients for weights in the 30 additional groups of "autos vs. motorcycles". The vertical lines are used to delineate the groups.

cancer data[4], which contains 8141 genes in 295 tumors (78 metastatic and 217 non-metastatic). As in (Jacob et al., 2009), we use the 300 genes that are most correlated to the output, and reduce the class imbalance by duplicating the positive samples twice. 50%, 30% and 20% of the data set are then randomly chosen for training, validation, and testing, respectively.

Table 3 shows the results averaged over 5 repetitions. As can be seen, both fused lasso and R-OSCAR can select more features than lasso for more accurate classification, while being able to group them and obtain much smaller dofs than the elastic net. Moreover, R-OSCAR is the most accurate among the four methods.

[4] `http://cbio.ensmp.fr/~ljacob/`

Table 3. Results on the breast cancer data.

	lasso	fused lasso	elastic net	R-OSCAR
test acc	65.93	72.41	71.83	**74.08**
dof	40.00	**17.00**	147.00	34.60
#nonzero feat.	40.00	217.2	147.00	103.80

5. Conclusion

In this paper, we used the accelerated gradient method to solve the optimization problem of the structured sparse OSCAR model. For the core projection step, we first studied the properties of its optimal solution and then showed that it can be solved by an iterative group merging algorithm. It is simple, easy to implement, and much more efficient than (1) the QP-based and SQP-based OSCAR solvers designed for OSCAR; and (2) the generic network flow algorithms for structured sparse models. Moreover, by rescaling the OSCAR solution with ridge regression, the unwanted over-penalty of large pairwise coefficient differences is alleviated. Experimental results show that it is a competitive regularizer for both regression and classification, but with the added ability of automatic feature grouping.

Acknowledgments

This research was supported in part by the Research Grants Council of the Hong Kong Special Administrative Region (Grant 615209). We thank Xiangneng Zeng for helpful discussions.

References

Bach, F. Structured sparsity-inducing norms through submodular functions. In *NIPS 24*. MIT Press, 2010.

Bach, F., Jenatton, R., Mairal, J., and Obozinski, G. Convex optimization with sparsity-inducing norms. In Sra, S., Nowozin, S., and Wright, S.J. (eds.), *Optimization for Machine Learning*. MIT Press, 2011.

Beck, A. and Teboulle, M. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

Bondell, H.D. and Reich, B.J. Simultaneous regression shrinkage, variable selection and clustering of predictors with OSCAR. *Biometrics*, 64(1):115, 2008.

Dettling, M. and Bühlmann, B. Finding predictive gene groups from microarray data. *Journal of Multivariate Analysis*, 90(1):106–131, 2004.

Duchi, J. and Singer, Y. Efficient online and batch learning using forward backward splitting. *Journal of Machine Learning Research*, 10:2873–2908, 2009.

Fujishige, S. *Submodular Functions and Optimization*. Elsevier, 2005.

Hastie, T., Tibshirani, R., Botstein, D., and Brown, P. Supervised harvesting of expression trees. *Genome Biology*, 2(1), 2001.

Jacob, L., Obozinski, G., and Vert, J.P. Group lasso with overlap and graph lasso. In *ICML*, pp. 433–440, Montreal, Canada, 2009.

Jenatton, R., Audibert, J.Y., and Bach, F. Structured variable selection with sparsity-inducing norms. Technical Report arXiv:0904.3523, 2009.

Jörnsten, R. and Yu, B. Simultaneous gene clustering and subset selection for sample classification via MDL. *Bioinformatics*, 19(9):1100–1109, 2003.

Mairal, J., Jenatton, R., Obozinski, G., and Bach, F. Network flow algorithms for structured sparsity. In *NIPS 23*, pp. 1558–1566. MIT Press, 2010a.

Mairal, J., Jenatton, R., Obozinski, G., and Bach, F. Network flow algorithms for structured sparsity. Technical Report arXiv:1008.5209, 2010b.

Nesterov, Y. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. In *Soviet Mathematics Doklady*, volume 27, pp. 372–376, 1983.

Nesterov, Y. Gradient methods for minimizing composite objective function. CORE Discussion Paper 2007/76, UniversitZ catholique de Louvain, 2007.

Shen, X. and Huang, H.C. Grouping pursuit through a regularization solution surface. *Journal of the American Statistical Association*, 105(490):727–739, 2010.

Tibshirani, R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58(1):267–288, 1996.

Tibshirani, R., Saunders, M., Rosset, S., Zhu, J., and Knight, K. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society, Series B*, 67(1):91–108, 2005.

Yuan, M. and Lin, Y. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B*, 68(1):49–67, 2006.

Zou, H. and Hastie, T. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society. Series B*, 67(2):301, 2005.