

An Efficient Proximal Gradient Method for General Structured Sparse Learning

Xi Chen

Qihang Lin

Seyoung Kim

Jaime G. Carbonell

Eric P. Xing

CARNEGIE MELLON UNIVERSITY
PITTSBURGH, PA 15213, USA

XICHEN@CS.CMU.EDU

QIHANGL@ANDREW.CMU.EDU

SSSYKIM@CS.CMU.EDU

JGC@CS.CMU.EDU

EPXING@CS.CMU.EDU

Editor:

Abstract

We study the problem of learning high dimensional regression models regularized by a structured-sparsity-inducing penalty that encodes prior structural information on either input or output sides. We consider two widely adopted types of such penalties as our motivating examples: 1) overlapping-group-lasso penalty, based on ℓ_1/ℓ_2 mixed-norm, and 2) graph-guided fusion penalty. For both types of penalties, due to their non-separability, developing an efficient optimization method has remained a challenging problem. In this paper, we propose a general optimization framework, called proximal gradient method, which can solve the structured sparse learning problems with a smooth convex loss and a wide spectrum of non-smooth and non-separable structured-sparsity-inducing penalties, including the overlapping-group-lasso and graph-guided fusion penalties. Our method exploits the structure of such penalties, decouples the non-separable penalty function via the dual norm, introduces its smooth approximation, and solves this approximation function. It achieves a convergence rate significantly faster than the standard first-order method, sub-gradient method, and is much more scalable than the most widely used method, namely interior-point method for second-order cone programming and quadratic programming formulations. The efficiency and scalability of our method are demonstrated on both simulated and real genetic datasets.

Keywords: Proximal Gradient Descent, Structured Sparsity, Overlapping Group Lasso, Graph-guided Fused Lasso

1. Introduction

The problem of high-dimensional sparse feature learning arises in many areas in science and engineering. In a typical setting, the input lies in a high-dimensional space, and one is interested in selecting a small number of input variables that influence the output. A popular approach to achieve this goal is to jointly optimize the fitness loss function with a non-smooth ℓ_1 -norm penalty (e.g., lasso (Tibshirani, 1996)) that shrinks the coefficients of the irrelevant input variables to zero. However, this approach is limited in that it treats each input as independent of each other and hence is incapable of capturing any structural

information among input variables. Recently, various extensions of the lasso penalty have been introduced to take advantage of the prior knowledge of the structure among inputs to encourage closely related inputs to be selected jointly (Yuan and Lin, 2006; Tibshirani and Saunders, 2005; Jenatton et al., 2009). Similar ideas have also been explored to leverage the output structures in multi-task learning, where one is interested in learning multiple related functional mappings from a common input space to multiple different outputs (Argyriou et al., 2008; Obozinski et al., 2009). In this case, the structure over the outputs is available as prior knowledge, and the closely related outputs according to this structure are encouraged to share a similar set of relevant inputs (Kim and Xing, 2010). Despite these progress, developing an efficient optimization method for solving the convex optimization problems resulting from the *structured-sparsity-inducing penalty functions* has remained a challenge. In particular, existing methods as reviewed below are limited to specialized forms of sparsity for relatively simple structures. In this paper, we focus on the problem of developing efficient optimization methods that can handle a broad set of structured-sparsity-inducing penalties with complex structures.

For structured-sparsity-inducing penalties with a relatively simple form of structures, such as in group lasso (Yuan and Lin, 2006) and fused lasso with a chain structure (Tibshirani and Saunders, 2005), efficient optimization methods have been available. For example, given the prior knowledge of non-overlapping group structure among input variables, group lasso uses a non-smooth mixed-norm penalty (e.g., ℓ_1/ℓ_2) to select groups of inputs as relevant to the output. Due to the separability of the group-lasso penalty (e.g., separability of the ℓ_2 norms, each corresponding to a group), a certain *projection step*, which involves a square Euclidean distance function and the penalty, can be computed in closed form. Based on this observation, a number of optimization approaches that use the projection as the key step have been applied to solve non-overlapping group lasso. Examples of such approaches include the first-order Nesterov’s method (Liu et al., 2009) and forward-backward splitting scheme (FOBOS) (Duchi and Singer, 2009). Fused lasso is another widely used extension of lasso which assumes that the inputs are ordered in a chain structure. By introducing a *fusion penalty* penalizing the difference between the coefficients for every two neighboring inputs, fused lasso encourages the coefficients for adjacent inputs to have the same values. Although the fused lasso penalty is not separable, the simple chain structure guarantees that the parameter for each input appears in only two terms of the penalty. Based on this fact, a simple and efficient pathwise coordinate descent approach for solving fused lasso has been proposed by Friedman et al. (2007).¹

Recently, in order to handle a more general class of structures such as trees or graphs over the inputs or outputs, various models that further extend group lasso and fused lasso have been proposed. For instance, generalizing the non-overlapping group structure assumed in the standard group lasso, Jenatton et al. (2009) proposed *overlapping group lasso*, which introduces overlaps among the groups so that each input can belong to multiple groups to allow capturing more complex and realistic prior knowledge on the structure. Another example of more general structured-sparsity-inducing penalties extends the chain-structured fused lasso to *graph-guided fused lasso*, where a fusion penalty is induced by every edge in a graph over inputs to enable closely related inputs as encoded in the graph to be

1. The author noted that there exists certain cases that the pathwise coordinate descent cannot converge to the exact solution.

selected jointly (Kim et al., 2009). In general, the existing fast optimization techniques for simpler structures cannot be applied to many of the non-trivial structured-sparsity-inducing penalties because of the non-separability of these penalties. Although in principle, generic optimization solvers such as the interior-point methods (IPM) could be used to solve either a second-order cone programming (SOCP) or a quadratic programming (QP) formulation of the problem, such approaches are computationally prohibitive even for problems of moderate size.

In this paper, we address the problem of efficient and scalable convex optimization for a very general class of structured-sparse-learning problems. We assume that the form of the penalty function includes (1) the standard lasso penalty (ℓ_1 -norm) for the individual feature level sparsity and (2) structured-sparsity-inducing penalty for incorporating prior knowledge on non-trivial structure over inputs or outputs. We propose a generic optimization framework called a *proximal gradient method* for dealing with a variety of structured-sparsity-inducing penalties that are both non-smooth and non-separable, using overlapping-group-lasso (Jenatton et al., 2009) and graph-guided fused lasso (Kim et al., 2009) as motivating examples. Although the overlapping-group-lasso penalty and graph-guided fusion penalty are seemingly very different, we show that it is possible to decouple the non-separable terms in both penalties via dual norm and reformulate them into a common form to which our optimization framework can be applied. Our approach is called a “proximal” gradient method because instead of optimizing the original problem directly, we introduce a smooth approximation of the structured-sparsity-inducing penalty and then apply the fast iterative shrinkage-thresholding algorithm (FISTA)(Beck and Teboulle, 2009). Our method can achieve $O(\frac{1}{\epsilon})$ convergence rate for a desired accuracy ϵ . In other words, it can find a solution β^t for minimizing f function after $t = O(\frac{1}{\epsilon})$ iterations such that $f(\beta^t) - f(\beta^*) \leq \epsilon$, where β^* is the optimal solution. There are several advantages in using our proximal gradient method:

- (a) Our method is a first-order method, as it uses only the gradient information. Thus, it is significantly more scalable than IPM for SOCP or QP formulations. In addition, since our method is gradient-based, it allows warm restarts, which makes it possible to efficiently solve the problem along the entire regularization path (Friedman et al., 2007).
- (b) Our optimization framework is ubiquitously applicable to any structured sparse learning problems with a smooth convex loss and a complex structured-sparsity-inducing penalty that is non-smooth and non-separable. The mixed-norm penalties as in overlapping group lasso and graph-guided fusion penalties are only examples of such penalties that we use to demonstrate our method.
- (c) Theoretically, our optimization method has a convergence rate of $O(\frac{1}{\epsilon})$, which is faster than the subgradient method with a convergence rate of $O(\frac{1}{\epsilon^2})$. In fact, for the optimization problems that we consider in this paper, there are no implementations of the subgradient method available in the literature.
- (d) Our method can be applied to both single-task (univariate-response) learning with a structure defined over inputs and multi-task learning with a structure defined over outputs.

- (e) Our method is easy to implement with only a few lines of MATLAB code.

The proposed method draws insights from two lines of earlier work in the literature that addresses optimization problems with a non-smooth penalty. The first approach (Nesterov, 2005) smoothes out the entire non-smooth term and then applies a gradient-based method to solve a completely smooth problem. However, it is precisely the *non-smoothness* of the penalty that leads to a sparse solution with the coefficients for irrelevant inputs set exactly to zeros. Therefore, although widely adopted in optimization field, this approach loses the merit of the non-smoothness of the penalty and hence cannot yield sparse solutions. The second approach (Beck and Teboulle, 2009; Nesterov, 2007) is widely applied in machine learning, generates sparse solutions via a so-called projection step (also called generalized gradient update step). However, this method requires that the non-smooth term is simple enough so that the projection step involving it can be solved in closed form. This type of approach simply cannot be applied to our problems due to the complicated and non-separable structured-sparsity-inducing penalty. The idea of our proximal gradient method combines these two different approaches: it smoothes out the complex structured-sparsity-inducing penalty while leaving the simple ℓ_1 -norm penalty as it is. Then we apply the second approach with projection step only involving the simple ℓ_1 -norm penalty. Therefore, our method will provide sparse solutions with the regression coefficients for irrelevant variables set exactly to zero.

The rest of this paper is organized as follows. In Section 2, we present the methods of overlapping group lasso and graph-guided fused lasso with group and graph structures encoded in the structured-sparsity-inducing penalties. In Section 3, we show how different structured-sparsity-inducing penalties can be reformulated into a common form. In Section 4, we present our proximal gradient method along with complexity results and discuss how this method can be applied to a logistic-regression loss. In Section 5, we present the generalization of our algorithm to the setting of the multi-task learning. Recent related works are discussed in Section 6. In Section 7, we present numerical results on both simulated and real datasets, followed by conclusions in Section 8. Throughout the paper, we will discuss overlapping-group-lasso and graph-guided fusion penalties in parallel to illustrate how our proximal gradient method can be used to solve the corresponding optimization problems.

2. Background: Linear Regression with Structured-sparsity-inducing Penalties

In this section, we provide a brief review of the high-dimensional linear regression model with structured-sparsity-inducing penalties, and discuss the overlapping-group-lasso penalty and graph-guided fusion penalty. We emphasize that our method can be applied to any smooth convex loss function (e.g., logistic loss) with a class of non-smooth and non-separable penalties. We focus on a single-task (univariate-response) learning problem, and return to the multi-task setting in Section 5.

Let $\mathbf{X} \in \mathbb{R}^{N \times J}$ denote the input data for N samples, where each sample lies in J dimensional space, and $\mathbf{y} \in \mathbb{R}^{N \times 1}$ be the output data. We assume a linear regression model, $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$, where $\boldsymbol{\beta}$ is the vector of length J for regression coefficients and $\boldsymbol{\epsilon}$ is the vector of length N for noise distributed as $N(0, \sigma^2 I_{N \times N})$. The standard lasso (Tibshirani,

1996) obtains a sparse estimate of the coefficients by solving the following optimization problem:

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^J} g(\boldsymbol{\beta}) + \lambda \|\boldsymbol{\beta}\|_1, \quad (1)$$

where $g(\boldsymbol{\beta}) \equiv \frac{1}{2} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2$ is the squared-error loss, $\|\boldsymbol{\beta}\|_1 \equiv \sum_{j=1}^J |\beta_j|$ is the ℓ_1 -norm penalty that encourages the solutions to be sparse, and λ is the regularization parameter that controls the sparsity level.

While the standard lasso penalty does not assume any structure among the input variables, recently various extensions of the lasso penalty have been proposed that incorporate the prior knowledge on the structure over the input variables to learn a joint sparsity pattern among related inputs. We broadly call the structured-sparsity-inducing penalty $\Omega(\boldsymbol{\beta})$ without assuming a specific form, and define the problem of learning a structured-sparsity pattern in the coefficients as follows:

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^J} f(\boldsymbol{\beta}) \equiv g(\boldsymbol{\beta}) + \Omega(\boldsymbol{\beta}) + \lambda \|\boldsymbol{\beta}\|_1. \quad (2)$$

In general, the structure over the input variables is assumed to be available in the form of groups or graphs, and the penalty $\Omega(\boldsymbol{\beta})$ is constructed based on this prior information. Then, the $\Omega(\boldsymbol{\beta})$ plays the role of encouraging the closely related inputs as described by the structure over the inputs to be selected jointly as relevant to the output by setting the corresponding regression coefficients to non-zero values. In other words, we encode the available structural information in $\Omega(\boldsymbol{\beta})$ to guide the learning process of sparse regression coefficients $\boldsymbol{\beta}$.

As examples of such penalties, in this paper, we consider two broad categories of penalties $\Omega(\boldsymbol{\beta})$ based on two different types of functional forms, namely overlapping-group-lasso penalty based on ℓ_1/ℓ_2 mixed-norm and graph-guided fusion penalty. As we discuss below, these two types of penalties cover a broad set of structured-sparsity-inducing penalties that have been introduced in the literature (Yuan and Lin, 2006; Jenatton et al., 2009; Kim and Xing, 2010; Zhao et al., 2009; Kim et al., 2009).

1. **Overlapping-group-lasso Penalty:** Let us assume that the set of groups of inputs $\mathcal{G} = \{g_1, \dots, g_{|\mathcal{G}|}\}$ is defined as a subset of the power set of $\{1, \dots, J\}$, and is available as prior knowledge. We note that the members (groups) of \mathcal{G} are allowed to overlap. Then the general overlapping group structures can be naturally encoded in the following structured-sparsity-inducing penalty:

$$\Omega(\boldsymbol{\beta}) \equiv \gamma \sum_{g \in \mathcal{G}} w_g \|\boldsymbol{\beta}_g\|_2, \quad (3)$$

where $\boldsymbol{\beta}_g \in \mathbb{R}^{|g|}$ is the subvector of $\boldsymbol{\beta}$ for the inputs in group g ; γ is the regularization parameter for structured sparsity; w_g is the predefined weight for group g ; and $\|\cdot\|_2$ is the vector ℓ_2 -norm. The ℓ_1/ℓ_2 mixed-norm penalty $\Omega(\boldsymbol{\beta})$ plays the role of setting all of the coefficients within each group to zero or non-zero values. One simple strategy for choosing the weight w_g is to set $w_g = \sqrt{|g|}$ (Yuan and Lin, 2006) so that the amount of penalization is adjusted by the size of each group.

We note that many of the structured-sparsity-inducing penalties in the current literature are a special case of (3). Examples include separated group structure (Yuan and Lin, 2006), tree structure (Zhao et al., 2009; Kim and Xing, 2010), where groups are defined for subtrees at each internal node, and graph structure, where each group is defined as two nodes of an edge.

2. **Graph-guided Fusion Penalty:** Let us assume the structure of J input variables is available as a graph G with a set of nodes $V = \{1, \dots, J\}$ and a set of edges E . Let $r_{ml} \in \mathbb{R}$ denote the weight of the edge $e = (m, l) \in E$, corresponding to the correlation between the two inputs for nodes m and l . Then, the graph-guided fusion penalty as defined below generalizes chain-structured fused-lasso penalty proposed by Tibshirani and Saunders (2005):

$$\Omega(\boldsymbol{\beta}) = \gamma \sum_{e=(m,l) \in E, m < l} \tau(r_{ml}) |\beta_m - \text{sign}(r_{ml})\beta_l|, \quad (4)$$

where $\tau(r)$ weights the fusion penalty for each edge $e = (m, l)$ such that β_m and β_l for highly correlated inputs with larger $|r_{ml}|$ receive a greater fusion effect. In this paper, we consider $\tau(r) = |r|$, but any monotonically increasing function of the absolute values of correlations can be used. The $\text{sign}(r_{ml})$ indicates that for two positively correlated nodes, the corresponding coefficients tend to be influence the output in the same direction, while for two negatively correlated nodes, the effects (β_m and β_l) take the opposite direction. Since this fusion effect is calibrated by the edge weight, the graph-guided fusion penalty in (4) encourages highly correlated inputs corresponding to a densely connected subnetwork in G to be jointly selected as relevant. We notice that if $r_{ml} = 1$ for all $e = (m, l)$, the penalty function in (4) reduces to:

$$\Omega(\boldsymbol{\beta}) = \gamma \sum_{e=(m,l) \in E, m < l} |\beta_m - \beta_l|. \quad (5)$$

The standard fused lasso penalty $\gamma \sum_{j=1}^{J-1} |\beta_{j+1} - \beta_j|$ is special case of (5), where the graph structure is confined to be a chain (Tibshirani and Saunders, 2005) and the widely used fused signal approximator refers to the simple case where the design matrix \mathbf{X} is orthogonal.

If the graph G is not available as prior knowledge, we can learn the graph from the input data by computing pairwise correlations based on \mathbf{x}_j 's and connecting two nodes with an edge if their correlation is above a given threshold ρ .

Although the optimization problem in (2) is convex, the main difficulty in optimizing (2) arises from the non-separability of $\boldsymbol{\beta}$ in the non-smooth penalty function $\Omega(\boldsymbol{\beta})$. As we show in the next section, although the two types of penalties are seemingly very different, we can reformulate both of them into the common form to which our proximal gradient method can be applied. The key idea in our approach is to decouple the non-separable structured-sparsity-inducing penalties into a simple linear transformation of $\boldsymbol{\beta}$ via the dual norm. Then, we introduce a smooth approximation of $\Omega(\boldsymbol{\beta})$ such that its gradient with respect to $\boldsymbol{\beta}$ can be easily calculated.

3. Reformulation of the Structured-sparsity-inducing Penalty

In this section, we show that despite the non-separability, by using the dual norm, both types of the structured-sparsity-inducing penalties in (3) and (4) can be decoupled once we reformulate them into the common form of a maximization problem over the auxiliary variables.

3.1 Overlapping-group-lasso Penalty

Since the dual norm of ℓ_2 -norm is also an ℓ_2 -norm, we can write $\|\beta_g\|_2$ as $\|\beta_g\|_2 = \max_{\|\alpha_g\|_2 \leq 1} \alpha_g^T \beta_g$, where $\alpha_g \in \mathbb{R}^{|g|}$ is the vector of auxiliary variables associated with β_g . Let $\alpha = [\alpha_{g_1}^T, \dots, \alpha_{g_{|\mathcal{G}|}}^T]^T$. Then, α is a vector of length $\sum_{g \in \mathcal{G}} |g|$ with domain $\mathcal{Q} \equiv \{\alpha \mid \|\alpha_g\|_2 \leq 1, \forall g \in \mathcal{G}\}$, where \mathcal{Q} is the Cartesian product of unit balls in Euclidean space and thus, a closed and convex set. We can rewrite the overlapping-group-lasso penalty in (3) as:

$$\Omega(\beta) = \gamma \sum_{g \in \mathcal{G}} w_g \max_{\|\alpha_g\|_2 \leq 1} \alpha_g^T \beta_g = \max_{\alpha \in \mathcal{Q}} \sum_{g \in \mathcal{G}} \gamma w_g \alpha_g^T \beta_g = \max_{\alpha \in \mathcal{Q}} \alpha^T C \beta, \quad (6)$$

where $C \in \mathbb{R}^{\sum_{g \in \mathcal{G}} |g| \times J}$ is a matrix defined as follows. The rows of C are indexed by all pairs of $(i, g) \in \{(i, g) \mid i \in g, i \in \{1, \dots, J\}\}$, the columns are indexed by $j \in \{1, \dots, J\}$, and each element of C is given as:

$$C_{(i,g),j} = \begin{cases} \gamma w_g & \text{if } i = j, \\ 0 & \text{otherwise.} \end{cases} \quad (7)$$

Then, we have $C\beta = [\gamma w_{g_1} \beta_{g_1}^T, \dots, \gamma w_{g_{|\mathcal{G}|}} \beta_{g_{|\mathcal{G}|}}^T]^T$.

Example. We give a concrete example of C . Assume $\beta \in \mathbb{R}^3$ (i.e., $J = 3$) with groups $\mathcal{G} = \{g_1 = \{1, 2\}, g_2 = \{2, 3\}\}$. Then, the matrix C is defined as follows:

$$\begin{array}{c} \begin{matrix} & j = 1 & j = 2 & j = 3 \\ i = 1 \in g_1 & \left(\begin{array}{ccc} \gamma w_{g_1} & 0 & 0 \\ 0 & \gamma w_{g_1} & 0 \\ 0 & \gamma w_{g_2} & 0 \\ 0 & 0 & \gamma w_{g_2} \end{array} \right) \end{matrix} \end{array}$$

Note that C is a highly sparse matrix with only a single non-zero element in each row and $\sum_{g \in \mathcal{G}} |g|$ non-zero elements in the entire matrix, and hence, can be stored with only a small amount of memory during the optimization procedure.

3.2 Graph-guided Fusion Penalty

First, we rewrite the graph-guided fusion penalty in (4) as follows:

$$\gamma \sum_{e=(m,l) \in E, m < l} \tau(r_{ml}) |\beta_m - \text{sign}(r_{ml}) \beta_l| \equiv \|C\beta\|_1,$$

where $C \in \mathbb{R}^{|E| \times J}$ is the edge-vertex incident matrix:

$$C_{e=(m,l),j} = \begin{cases} \gamma \cdot \tau(r_{ml}) & \text{if } j = m \\ -\gamma \cdot \text{sign}(r_{ml})\tau(r_{ml}) & \text{if } j = l \\ 0 & \text{otherwise.} \end{cases} \quad (8)$$

Again, we note that C is a highly sparse matrix with $2 \cdot |E|$ non-zero elements.

Since the dual norm of the ℓ_∞ -norm is the ℓ_1 -norm, we can further rewrite the graph-guided fusion penalty as:

$$\|C\boldsymbol{\beta}\|_1 \equiv \max_{\|\boldsymbol{\alpha}\|_\infty \leq 1} \boldsymbol{\alpha}^T C\boldsymbol{\beta}, \quad (9)$$

where $\boldsymbol{\alpha} \in \mathcal{Q} = \{\boldsymbol{\alpha} \mid \|\boldsymbol{\alpha}\|_\infty \leq 1, \boldsymbol{\alpha} \in \mathbb{R}^{|E|}\}$ is a vector of auxiliary variables associated with $\|C\boldsymbol{\beta}\|_1$, and $\|\cdot\|_\infty$ is the ℓ_∞ -norm defined as the maximum absolute value of all entries in the vector.

Remark 1 *We notice that the reformulation in (9) can be applied to not only the graph-guided fusion penalty but also any penalties in the form of $\Omega(\boldsymbol{\beta}) = \|C\boldsymbol{\beta}\|_1$, where C is any matrix with J columns and $\|C\boldsymbol{\beta}\|_1$ is the ℓ_1 -norm of a linear mapping of $\boldsymbol{\beta}$. Thus, our optimization method can be applied to a more general form of penalties than simply the graph-guided fusion penalty.*

4. Proximal Gradient Method

4.1 Smooth Approximation of Structured-sparsity-inducing Penalty

In the previous section, we showed that the structured-sparsity-inducing penalty $\Omega(\boldsymbol{\beta})$ for both overlapping-group-lasso and graph-guided fusion penalty can be reformulated as:

$$\Omega(\boldsymbol{\beta}) = \max_{\boldsymbol{\alpha} \in \mathcal{Q}} \boldsymbol{\alpha}^T C\boldsymbol{\beta}. \quad (10)$$

The formulation in (10) is still a non-smooth function of $\boldsymbol{\beta}$, and this makes the optimization challenging. To tackle this problem, in this section, we introduce an auxiliary quadratic function to construct a smooth approximation of (10) using the idea in Nesterov (2005). Our smooth approximation function is given as follows:

$$f_\mu(\boldsymbol{\beta}) = \max_{\boldsymbol{\alpha} \in \mathcal{Q}} (\boldsymbol{\alpha}^T C\boldsymbol{\beta} - \mu d(\boldsymbol{\alpha})), \quad (11)$$

where μ is the positive smoothness parameter and $d(\boldsymbol{\alpha})$ is defined as $\frac{1}{2}\|\boldsymbol{\alpha}\|_2^2$. The original penalty term can be viewed as $f_\mu(\boldsymbol{\beta})$ with $\mu = 0$ (i.e., $f_0(\boldsymbol{\beta}) = \max_{\boldsymbol{\alpha} \in \mathcal{Q}} \boldsymbol{\alpha}^T C\boldsymbol{\beta}$). It is easy to see that $f_\mu(\boldsymbol{\beta})$ is a lower bound of $f_0(\boldsymbol{\beta})$. In order to bound the gap between $f_\mu(\boldsymbol{\beta})$ and $f_0(\boldsymbol{\beta})$, let $D = \max_{\boldsymbol{\alpha} \in \mathcal{Q}} d(\boldsymbol{\alpha})$. Then, it is easy to verify that this gap is given by $f_0(\boldsymbol{\beta}) - f_\mu(\boldsymbol{\beta}) \leq \mu D$, where $D = |\mathcal{G}|/2$ for the overlapping-group-lasso penalty and $D = |E|/2$ for the graph-guided fusion penalty. From Theorem 1 as presented below, we know that $f_\mu(\boldsymbol{\beta})$ is a smooth function for any $\mu > 0$. Therefore, $f_\mu(\boldsymbol{\beta})$ can be viewed as a smooth approximation of $f_0(\boldsymbol{\beta})$ with the maximum gap of μD , and the μ controls the gap between $f_\mu(\boldsymbol{\beta})$ and $f_0(\boldsymbol{\beta})$. Given the desired accuracy ϵ , the convergence result in the next section suggests $\mu = \frac{\epsilon}{2D}$ to achieve the best convergence rate.

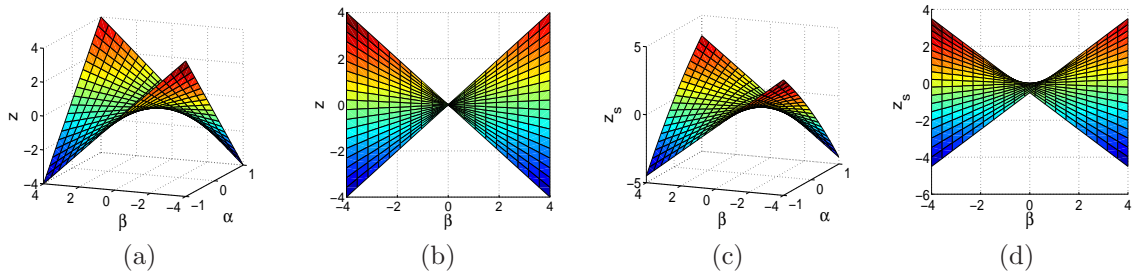


Figure 1: A geometric illustration of the smoothness of $f_\mu(\beta)$. (a) The 3-D plot of $z(\alpha, \beta)$, (b) the projection of (a) onto the β - z space, (c) the 3-D plot of $z_s(\alpha, \beta)$, and (d) the projection of (c) onto the β - z space.

Now, we present the key theorem that $f_\mu(\beta)$ is smooth in β with a simple form of gradient. This theorem is also stated by Nesterov (2005) but without a proof of smoothness property and a derivation of the gradient. In this paper, we provide a simple proof based on Fenchel Conjugate and properties of subdifferential. Intuitively, the strong convexity of $d(\alpha)$ leads to the smoothness of $f_\mu(\beta)$.

Theorem 2 For any $\mu > 0$, $f_\mu(\beta)$ is a convex and continuously-differentiable function in β , and the gradient of $f_\mu(\beta)$ takes the following form:

$$\nabla f_\mu(\beta) = C^T \alpha^*, \quad (12)$$

where α^* is the optimal solution to (11). Moreover, the gradient $\nabla f_\mu(\beta)$ is Lipschitz continuous with the Lipschitz constant $L_\mu = \frac{1}{\mu} \|C\|^2$, where $\|C\|$ is a special norm of C defined as $\|C\| \equiv \max_{\|\mathbf{v}\|_2 \leq 1} \|C\mathbf{v}\|_2$.

Proof The proof of this theorem is presented in Appendix. ■

To provide insights on why $f_\mu(\beta)$ is a smooth function as Theorem 1 suggests, in Figure 1, we show a geometric illustration for the case of one-dimensional parameter (i.e., $\beta \in \mathbb{R}$). For the sake of simplicity, we assume that μ and C are set to 1. First, we show geometrically that $f_0(\beta) = \max_{\alpha \in [-1, 1]} z(\alpha, \beta)$, where $z(\alpha, \beta) \equiv \alpha\beta$, is a non-smooth function. The three-dimensional plot for $z(\alpha, \beta)$ with α restricted to $[-1, 1]$ is shown in Figure 1(a). We project the surface in Figure 1(a) onto the β - z space as shown in Figure 1(b). For each β , the value of $f_0(\beta)$ is the highest point along the z -axis since we maximize over α in $[-1, 1]$. We can see that $f_0(\beta)$ is composed of two segments with a sharp point at $\beta = 0$. Now, we introduce the auxiliary function, and let $z_s(\alpha, \beta) \equiv \alpha\beta - \frac{1}{2}\alpha^2$ and $f_\mu(\beta) = \max_{\alpha \in [-1, 1]} z_s(\alpha, \beta)$. The three-dimensional plot for $z_s(\alpha, \beta)$ with α restricted to $[-1, 1]$ is shown in Figure 1(c). Similarly, we project the surface in Figure 1(c) onto the β - z_s space as shown in Figure 1(d). For fixed β , the value of $f_\mu(\beta)$ is the highest point along the z -axis. In Figure 1(d), we can see that $f_\mu(\beta)$ is composed of three parts: (i) a line with slope -1 when $\beta < -1$, (ii) a line with slope 1 when $\beta > 1$, and (iii) a quadratic function when $-1 \leq \beta \leq 1$. By introducing an auxiliary quadratic function, we remove the sharp point at $\beta = 0$ and $f_\mu(\beta)$ becomes a smooth function.

To compute the $\nabla f_\mu(\boldsymbol{\beta})$ and L_μ , we need to know $\boldsymbol{\alpha}^*$ and $\|C\|$. We present the closed-form equations for $\boldsymbol{\alpha}^*$ and $\|C\|$ for the overlapping-group-lasso penalty and graph-guided fusion penalty in the following propositions.

1. Overlapping-group-lasso Penalty

Proposition 3 *Let $\boldsymbol{\alpha}^*$, which is composed of $\{\boldsymbol{\alpha}_g^*\}_{g \in \mathcal{G}}$, be the optimal solution to (11) for group-lasso penalty with overlapping groups in (3). For any $g \in \mathcal{G}$,*

$$\boldsymbol{\alpha}_g^* = S\left(\frac{\gamma w_g \boldsymbol{\beta}_g}{\mu}\right),$$

where S is the shrinkage operator defined for any vector \mathbf{u} as follows:

$$S(\mathbf{u}) = \begin{cases} \frac{\mathbf{u}}{\|\mathbf{u}\|_2} & \|\mathbf{u}\|_2 > 1, \\ \mathbf{u} & \|\mathbf{u}\|_2 \leq 1. \end{cases}$$

Proof Taking the derivative of (11) with respect to $\boldsymbol{\alpha}$ and setting it to zero, we obtain $\boldsymbol{\alpha}_g = \frac{\gamma w_g \boldsymbol{\beta}_g}{\mu}$. We project the solution onto the \mathcal{Q} to obtain the optimal solution. ■

Proposition 4

$$\|C\| = \gamma \max_{j \in \{1, \dots, J\}} \sqrt{\sum_{g \in \mathcal{G} \text{ s.t. } j \in g} (w_g)^2}. \quad (13)$$

Proof See the proof in Appendix. ■

2. Graph-guided Fusion Penalty

Proposition 5 *Let $\boldsymbol{\alpha}^*$ be the optimal solution of (11) for graph-guided fusion penalty in (4). Then, we have:*

$$\boldsymbol{\alpha}^* = S\left(\frac{C\boldsymbol{\beta}}{\mu}\right),$$

where S is the shrinkage operator defined as follows:

$$S(x) = \begin{cases} x, & \text{if } -1 \leq x \leq 1 \\ 1, & \text{if } x > 1 \\ -1, & \text{if } x < -1. \end{cases}$$

For any vector $\boldsymbol{\alpha}$, $S(\boldsymbol{\alpha})$ is defined as applying S on each and every entry of $\boldsymbol{\alpha}$.

Proof The proof for this proposition is similar to the one for Proposition 3. ■

Proposition 6 $\|C\|$ is upper-bounded by $\sqrt{2\gamma^2 \max_{j \in V} d_j}$, where

$$d_j = \sum_{e \in E \text{ s.t. } e \text{ incident on } j} (\tau(r_e))^2 \quad (14)$$

for $j \in V$ in graph G , and this bound is tight.

Proof See the proof in Appendix. ■

Note that when $\tau(r_e) = 1$ for all $e \in E$, d_j is simply the degree of the node j .

4.2 Proximal Gradient Descent

Given the smooth approximation of the non-smooth structured-sparsity-inducing penalties as presented in the previous section, now, we apply the fast iterative shrinkage-thresholding algorithm (FISTA) (Beck and Teboulle, 2009), using the gradient information in Theorem 2. We substitute the penalty term $\Omega(\boldsymbol{\beta})$ in (2) with its smooth approximation $f_\mu(\boldsymbol{\beta})$ to obtain the following optimization problem:

$$\min_{\boldsymbol{\beta}} \tilde{f}(\boldsymbol{\beta}) \equiv g(\boldsymbol{\beta}) + f_\mu(\boldsymbol{\beta}) + \lambda \|\boldsymbol{\beta}\|_1. \quad (15)$$

Let

$$h(\boldsymbol{\beta}) = g(\boldsymbol{\beta}) + f_\mu(\boldsymbol{\beta}) = \frac{1}{2} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + f_\mu(\boldsymbol{\beta}).$$

According to Theorem 2, the gradient of $h(\boldsymbol{\beta})$ is given as:

$$\nabla h(\boldsymbol{\beta}) = \mathbf{X}^T(\mathbf{X}\boldsymbol{\beta} - \mathbf{y}) + C^T \boldsymbol{\alpha}^*. \quad (16)$$

Moreover, $\nabla h(\boldsymbol{\beta})$ is Lipschitz-continuous with the Lipschitz constant:

$$L = \lambda_{\max}(\mathbf{X}^T \mathbf{X}) + L_\mu = \lambda_{\max}(\mathbf{X}^T \mathbf{X}) + \frac{\|C\|^2}{\mu}, \quad (17)$$

where $\lambda_{\max}(\mathbf{X}^T \mathbf{X})$ is the largest eigenvalue of $(\mathbf{X}^T \mathbf{X})$.

Since $f(\boldsymbol{\beta})$ only involves a very simple non-smooth part (i.e., the ℓ_1 -norm penalty), we can adopt FISTA (Beck and Teboulle, 2009) to minimize $\tilde{f}(\boldsymbol{\beta})$ as shown in Algorithm 1.

We briefly discuss the reason why the step in (18) is called ‘‘generalized gradient update step’’. For any smooth convex function h , it is well-known that the traditional gradient descent step with $1/L$ as the step-size is equivalent to solving a quadratic approximation (upper bound) of h :

$$\boldsymbol{\beta}^{t+1} = \mathbf{w}^t - \frac{1}{L} \nabla h(\mathbf{w}^t) \iff \boldsymbol{\beta}^{t+1} = \arg \min_{\boldsymbol{\beta}} h(\mathbf{w}^t) + \langle \boldsymbol{\beta} - \mathbf{w}^t, \nabla h(\mathbf{w}^t) \rangle + \frac{L}{2} \|\boldsymbol{\beta} - \mathbf{w}^t\|_2^2, \quad (19)$$

Algorithm 1 Proximal Gradient Method for Learning Structured Sparsity

Input: $\mathbf{X}, \mathbf{y}, C, \boldsymbol{\beta}^0$, desired accuracy ϵ .

Initialization: set $\mu = \frac{\epsilon}{2D}$, $\theta_0 = 1$, $\mathbf{w}^0 = \boldsymbol{\beta}^0$.

Iterate For $t = 0, 1, 2, \dots$, until convergence of $\boldsymbol{\beta}^t$:

1. Compute $\nabla h(\mathbf{w}^t)$ according to (16).
2. Perform the generalized gradient update step:

$$\boldsymbol{\beta}^{t+1} = \arg \min_{\boldsymbol{\beta}} h(\mathbf{w}^t) + \langle \boldsymbol{\beta} - \mathbf{w}^t, \nabla h(\mathbf{w}^t) \rangle + \lambda \|\boldsymbol{\beta}\|_1 + \frac{L}{2} \|\boldsymbol{\beta} - \mathbf{w}^t\|_2^2 \quad (18)$$

3. Set $\theta_{t+1} = \frac{2}{t+3}$.
4. Set $\mathbf{w}^{t+1} = \boldsymbol{\beta}^{t+1} + \frac{1-\theta_t}{\theta_t} \theta_{t+1} (\boldsymbol{\beta}^{t+1} - \boldsymbol{\beta}^t)$.

Output: $\hat{\boldsymbol{\beta}} = \boldsymbol{\beta}^{t+1}$.

where $h(\mathbf{w}^t) + \langle \boldsymbol{\beta} - \mathbf{w}^t, \nabla h(\mathbf{w}^t) \rangle$ is a linear approximation of h at $\boldsymbol{\beta}$. Based on (19), the generalized gradient update step adds $\lambda \|\boldsymbol{\beta}\|_1$ to enforce the solution to be sparse and it can be computed in closed form: rewriting (18), we obtain:

$$\boldsymbol{\beta}^{t+1} = \arg \min_{\boldsymbol{\beta}} \frac{1}{2} \|\boldsymbol{\beta} - (\mathbf{w}^t - \frac{1}{L} \nabla h(\mathbf{w}^t))\|_2^2 + \frac{\lambda}{L} \|\boldsymbol{\beta}\|_1.$$

Let $\mathbf{v} = (\mathbf{w}^t - \frac{1}{L} \nabla h(\mathbf{w}^t))$, the closed-form solution for $\boldsymbol{\beta}^{t+1}$ is given as in the next proposition.

Proposition 7 *The closed-form solution of*

$$\min_{\boldsymbol{\beta}} \frac{1}{2} \|\boldsymbol{\beta} - \mathbf{v}\|_2^2 + \frac{\lambda}{L} \|\boldsymbol{\beta}\|_1$$

can be obtained by the soft-thresholding operation:

$$\beta_j = \text{sign}(v_j) \max(0, |v_j| - \frac{\lambda}{L}), \quad j = 1, \dots, J. \quad (20)$$

Proof The problem can be decomposed into J independent subproblems, each of which is given as $\min_{\beta_j} \frac{1}{2} \|\beta_j - v_j\|_2^2 + \frac{\lambda}{L} |\beta_j|$. For each subproblem, taking the subgradient with respect to v_j and requiring that 0 belongs to the subgradient yield the closed-form solution in (20). ■

A notable advantage of utilizing the generalized gradient update step with the simple ℓ_1 -norm penalty is that it can provide us with the sparse solutions, where the coefficients for irrelevant inputs are set exactly to zero, due to the soft-thresholding operation in (20). At convergence, the values of $\boldsymbol{\beta}$ for the irrelevant groups in the group-lasso penalty and edges in the graph-guided fusion penalty will become sufficiently close to zeros, and the soft-thresholding operation will truncate them exactly to zeros, leading to sparse solutions.

Table 1: Comparison of Time Complexity

	Overlapping Group Lasso	Graph-guided Fused Lasso
Proximal Gradient	$O(J^2 + \sum_{g \in \mathcal{G}} g)$	$O(J^2 + E)$
IPM for SOCP	$O\left((J + \mathcal{G})^2(N + \sum_{g \in \mathcal{G}} g)\right)$	$O\left((J + E)^2(N + J + E)\right)$

Remark 8 *If we directly apply FISTA to the original problem (2), the generalized gradient update step will involve $\Omega(\boldsymbol{\beta})$ and hence does not have a closed-form solution. This is the challenge that we circumvent via smoothing.*

Remark 9 *Our proximal gradient algorithm is a general approach that can be applied to problems with any kind of smooth convex loss functions and structured-sparse-inducing penalties that can be rewritten in the form of $\max_{\boldsymbol{\alpha}} \boldsymbol{\alpha}^T C \boldsymbol{\beta}$.*

4.3 Convergence Rate and Time Complexity

Although we optimize the approximation function $\tilde{f}(\boldsymbol{\beta})$ rather than optimizing $f(\boldsymbol{\beta})$ directly, it can be proven that the $\tilde{\boldsymbol{\beta}}$ obtained from Algorithm 1 is sufficiently close to the optimal solution $\boldsymbol{\beta}^*$ to the original optimization problem in (2). We present the convergence rate of Algorithm 1 in the next theorem.

Theorem 10 *Let $\boldsymbol{\beta}^*$ be the optimal solution to (2) and $\boldsymbol{\beta}^t$ be the approximate solution at the t -th iteration in Algorithm 1. If we require $f(\boldsymbol{\beta}^t) - f(\boldsymbol{\beta}^*) \leq \epsilon$ and set $\mu = \frac{\epsilon}{2D}$, then, the number of iterations t is upper-bounded by*

$$\sqrt{\frac{4\|\boldsymbol{\beta}^* - \boldsymbol{\beta}^0\|_2^2}{\epsilon} \left(\lambda_{\max}(\mathbf{X}^T \mathbf{X}) + \frac{2D\|C\|^2}{\epsilon} \right)}. \quad (21)$$

The key idea behind the proof of this theorem is to decompose $f(\boldsymbol{\beta}^t) - f(\boldsymbol{\beta}^*)$ into three parts: (i) $f(\boldsymbol{\beta}^t) - \tilde{f}(\boldsymbol{\beta}^t)$, (ii) $\tilde{f}(\boldsymbol{\beta}^t) - \tilde{f}(\boldsymbol{\beta}^*)$, and (iii) $\tilde{f}(\boldsymbol{\beta}^*) - f(\boldsymbol{\beta}^*)$. (i) and (iii) can be bounded by the gap of the approximation μD . (ii) only involves the function \tilde{f} and can be upper bounded by $O(\frac{1}{t^2})$ as shown in Beck and Teboulle (2009). We obtain (21) by balancing these three terms. The details of the proof are presented in Appendix. According to Theorem 10, Algorithm 1 converges in $O(\frac{\sqrt{2D}}{\epsilon})$ iterations, which is much faster than the subgradient method with the convergence rate of $O(\frac{1}{\epsilon^2})$. Note that the convergence rate depends on D through the term $\sqrt{2D}$, and the D depends on the problem size.

As for the time complexity, assuming that we can pre-compute and store $\mathbf{X}^T \mathbf{X}$ and $\mathbf{X}^T \mathbf{y}$ with the time complexity of $O(J^2 N)$, the main computational cost in each iteration comes from calculating the gradient $\nabla h(\mathbf{w}_t)$. The per-iteration time complexity of Algorithm 1 as compared to IPM for SOCP according to Lobo et al. (1998) is presented in Table 1.

Remark 11 *According to Table 1, although IPM converges in fewer iterations (i.e., $\log(\frac{1}{\epsilon})$), its per-iteration complexity is higher by orders of magnitude than that of our method. For example, for the overlapping-group-lasso penalty, the per-iteration complexity of our method*

is $O(J^2 + \sum_{g \in \mathcal{G}} |g|)$, which depends on the sum of J^2 and $\sum_{g \in \mathcal{G}} |g|$, while that of IPM for SOCP is at least the product of J^2 and $\sum_{g \in \mathcal{G}} |g|$. For the graph-guided fusion penalty, the per-iteration complexity of our method is linear in $|E|$, while that of IPM for SOCP is cubic in $|E|$. Therefore, our method is much more scalable for large-scale problems. In addition to the higher time complexity, each IPM iteration of SOCP requires significantly more memory to store the Newton linear system.

Remark 12 *If we can pre-compute and store $\mathbf{X}^T \mathbf{X}$, the per-iteration time complexity of our method is independent of sample size N as shown in Table 1. If J is very large, $\mathbf{X}^T \mathbf{X}$ may not fit into memory. If this is the case, instead of pre-computing and storing $\mathbf{X}^T \mathbf{X}$, we can compute $\mathbf{X}^T (\mathbf{X} \mathbf{w}^t)$ in each iteration. Then, the per-iteration time complexity will increase by a factor of N , but this is still less than that of IPM for SOCP.*

4.4 Logistic Loss for Classification Problems

As we stated in Remark 9, our method is a general approach that can be applied to any convex smooth loss. For classification problems with each output data $y_i \in \{0, 1\}$, it is more natural to adopt the logistic loss:

$$g(\boldsymbol{\beta}) = - \sum_{i=1}^N \log \mathbb{P}(y_i | \mathbf{x}_i) = \sum_{i=1}^N [\log(1 + \exp(\boldsymbol{\beta}^T \mathbf{x}_i)) - y_i \boldsymbol{\beta}^T \mathbf{x}_i], \quad (22)$$

with the gradient

$$\nabla g(\boldsymbol{\beta}) = \sum_{i=1}^N \mathbf{x}_i \left(\frac{\exp(\boldsymbol{\beta}^T \mathbf{x}_i)}{1 + \exp(\boldsymbol{\beta}^T \mathbf{x}_i)} - y_i \right). \quad (23)$$

We can directly apply Algorithm 1 to solve the logistic regression with the structured-sparsity-inducing penalty, where the gradient of $h(\boldsymbol{\beta}) = g(\boldsymbol{\beta}) + f_\mu(\boldsymbol{\beta})$ is $\nabla g(\boldsymbol{\beta}) + C^T \boldsymbol{\alpha}^*$. According to the next proposition, the Lipschitz constant of $\nabla g(\boldsymbol{\beta})$ is simply $\sum_{i=1}^N \|\mathbf{x}_i\|^2$.

Proposition 13 *The Lipschitz constant for $\nabla g(\boldsymbol{\beta})$ as defined in (23) is $\sum_{i=1}^N \|\mathbf{x}_i\|^2$.*

Proof See the proof in Appendix. ■

5. Extensions for Multi-task Learning

The structured-sparsity-inducing penalties as discussed in the previous section can be similarly used in the multi-task regression setting. Depending on the types of structures over the *outputs* that are available as prior information, different types of penalties including overlapping-group-lasso penalty and graph-guided fusion penalty have been applied to learn the shared sparsity pattern across multiple tasks (Kim and Xing, 2010; Kim et al., 2009). For example, in genetic association analysis, where the goal is to discover few genetic variants or single nucleotide polymorphisms (SNPs) out of millions of SNPs (inputs) that influence phenotypes (outputs) such as gene expression measurements, the correlation structure of phenotypes can be naturally represented as a graph, which can be used to guide

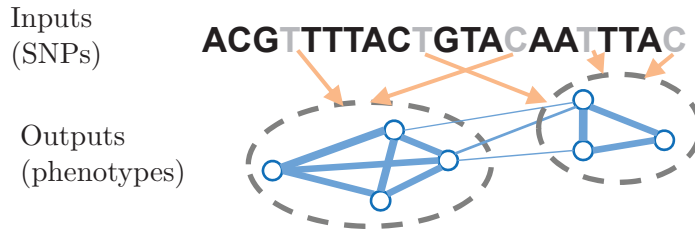


Figure 2: Illustration of the multi-task regression with graph structure on outputs.

the selection of SNPs as shown in Figure 2. Then, the graph-guided fusion penalty can be used to identify SNPs that are relevant jointly to multiple related phenotypes.

While in the multi-task regression problem, we encounter the same difficulties of optimizing with non-smooth and non-separable penalties as in the previous section, our proximal gradient method can be extended to this problem in a straightforward manner. For completeness, in this section, we briefly discuss how our method can be applied to the multi-task regression with structured-sparsity-inducing penalties. For the ease of illustration, we discuss cases where different tasks (outputs) share the same input matrix.

5.1 Multi-task Linear Regression and Structured-sparsity-inducing Penalty

Let $\mathbf{X} \in \mathbb{R}^{N \times J}$ denote the matrix of input data for J inputs and $\mathbf{Y} \in \mathbb{R}^{N \times K}$ denote the matrix of output data for K outputs over N samples. We assume a linear regression model for each of the k -th task: $\mathbf{y}_k = \mathbf{X}\boldsymbol{\beta}_k + \boldsymbol{\epsilon}_k$, $\forall k = 1, \dots, K$, where $\boldsymbol{\beta}_k = [\beta_{1k}, \dots, \beta_{Jk}]^T$ is the regression coefficient vector for the k -th task and $\boldsymbol{\epsilon}_k$ is Gaussian noise. Let $\mathbf{B} = [\boldsymbol{\beta}_1, \dots, \boldsymbol{\beta}_K] \in \mathbb{R}^{J \times K}$ be the matrix of regression coefficients for all of the K tasks. Then, the multi-task structured sparse regression problem can be naturally formulated as the following optimization problem:

$$\min_{\mathbf{B} \in \mathbb{R}^{J \times K}} f(\mathbf{B}) \equiv \frac{1}{2} \|\mathbf{Y} - \mathbf{X}\mathbf{B}\|_F^2 + \Omega(\mathbf{B}) + \lambda \|\mathbf{B}\|_1, \quad (24)$$

where $\|\cdot\|_F$ denotes the matrix Frobenius norm, $\|\cdot\|_1$ denotes the matrix entry-wise ℓ_1 norm, and $\Omega(\mathbf{B})$ is a structured-sparsity-inducing penalty with the structure over tasks.

1. **Multi-task Overlapping-group-lasso Penalty:** We define the overlapping-group-lasso penalty for the multi-task regression as follows:

$$\Omega(\mathbf{B}) \equiv \gamma \sum_{j=1}^J \sum_{g \in \mathcal{G}} w_g \|\boldsymbol{\beta}_{jg}\|_2, \quad (25)$$

where $\mathcal{G} = \{g_1, \dots, g_{|\mathcal{G}|}\}$ is a subset of the power set of $\{1, \dots, K\}$ and $\boldsymbol{\beta}_{jg}$ is the vector of regression coefficients $\{\beta_{jk}, k \in g\}$. Both ℓ_1/ℓ_2 mixed-norm penalty for multi-task regression (Obozinski et al., 2009) and tree-guided group-lasso penalty (Kim and Xing, 2010) are special cases of (25).

2. **Multi-task Graph-guided Fusion Penalty:** Assuming the graph structure over the K outputs is given as G with a set of nodes $V = \{1, \dots, K\}$ and a set of edges E , the

Table 2: Comparison of Time Complexity for Multi-task Regression

	Overlapping Group Lasso	Graph-guided Fused Lasso
Proximal Gradient	$O(J^2K + J \sum_{g \in \mathcal{G}} g)$	$O(J^2K + J E)$
IPM for SOCP	$O\left(J^2(K + \mathcal{G})^2(KN + J(\mathcal{G} + \sum_{g \in \mathcal{G}} g))\right)$	$O(J^2(K + E)^2(KN + JK + J E))$

graph-guided fusion penalty for multi-task regression is given as:

$$\Omega(\mathbf{B}) = \gamma \sum_{e=(m,l) \in E} \tau(r_{ml}) \sum_{j=1}^J |\beta_{jm} - \text{sign}(r_{ml})\beta_{jl}|. \quad (26)$$

5.2 Proximal Gradient Descent

Using the similar techniques in Section 3, $\Omega(\mathbf{B})$ can be reformulated as:

$$\Omega(\mathbf{B}) = \max_{\mathbf{A} \in \mathcal{Q}} \langle C\mathbf{B}^T, \mathbf{A} \rangle, \quad (27)$$

where $\langle \mathbf{U}, \mathbf{V} \rangle \equiv \text{Tr}(\mathbf{U}^T \mathbf{V})$ denotes a matrix inner product. C is constructed in the similar way as in (7) or (8) just by replacing the index of the input variables with the output variables, and \mathbf{A} is the matrix of auxiliary variables.

Then we introduce the smooth approximation of (27):

$$f_\mu(\mathbf{B}) = \max_{\mathbf{A} \in \mathcal{Q}} (\langle C\mathbf{B}^T, \mathbf{A} \rangle - \mu d(\mathbf{A})), \quad (28)$$

where $d(\mathbf{A}) \equiv \frac{1}{2} \|\mathbf{A}\|_F^2$. Following a proof strategy similar to that in Theorem 2, we can show that $f_\mu(\mathbf{B})$ is convex and smooth with gradient $\nabla f_\mu(\mathbf{B}) = (\mathbf{A}^*)^T C$, where \mathbf{A}^* is the optimal solution to (28). For overlapping-group-lasso penalty, \mathbf{A} is composed of $\alpha_{jg}^* = S(\frac{\lambda w_g \beta_{jg}}{\mu})$; for graph-guided fusion penalty, $\mathbf{A} = S(\frac{C\mathbf{B}^T}{\mu})$. In addition, $\nabla f_\mu(\mathbf{B})$ is Lipschitz continuous with the Lipschitz constant $L_\mu = \|C\|^2/\mu$, where $\|C\| \equiv \max_{\|\mathbf{V}\|_F \leq 1} \|C\mathbf{V}^T\|_F$. Similar to proposition 4 and 6, we can show that for overlapping-group-lasso penalty $\|C\| = \gamma \max_{k \in \{1, \dots, K\}} \sqrt{\sum_{g \in \mathcal{G} \text{ s.t. } k \in g} (w_g)^2}$ and for graph-guided fusion penalty, $\|C\|$ is upper bounded by $\sqrt{2\gamma^2 \max_{j \in V} d_k}$, where d_k is defined in (14).

By substituting $\Omega(\mathbf{B})$ in (24) with $f_\mu(\mathbf{B})$, we can adopt Algorithm 1 to solve (24) with convergence rate of $O(\frac{1}{\epsilon})$ iterations. The per-iteration time complexity compared to IPM for SOCP is shown in Table 2. It can be seen that our method is much more efficient than IPM for SOCP.

6. Related Work

Recently, first-order approach such as in Nesterov's method (Nesterov, 2007), FISTA (Beck and Teboulle, 2009), and forward-backward splitting (FOBOS) (Duchi and Singer, 2009) has been widely adopted to solve optimization problems with a convex loss and non-smooth penalty. However, most of the existing works dealt with relatively simple and

well-separated non-smooth penalties (e.g., ℓ_1 -norm, ℓ_1/ℓ_2 mixed-norm penalty with non-overlapping groups, ℓ_1/ℓ_2 mixed-norm in multi-task regression) so that the generalized gradient update step (or so-called projection step) with the original form of the penalty can be obtained in a closed form. In this case, the first-order approach can achieve the optimal convergence rate $O(\frac{1}{\sqrt{\epsilon}})$. However, the structure of the penalties in our problem is very complex, and such a projection step does not have a closed-form solution. This is the challenge that we circumvent via smoothing in our proximal gradient method.

The idea of smoothing the non-smooth function was initiated by Nesterov (2005). Since the method by Nesterov (2005) works only for smooth problems, it has to smooth out the entire non-smooth penalty including the ℓ_1 -norm. However, it is precisely the *non-smoothness* of the penalty that leads to exact zeros in optimal solutions. Therefore, the method in Nesterov (2005) cannot yield a sparse solution. Moreover, their algorithm requires that β is bounded, and that the number of iterations is pre-defined, although these conditions are all impractical for real applications. Instead, our approach leads to the sparse solutions with the coefficients for irrelevant variables exactly set to zero, and has a simple and novel derivation of the convergence rate. As for the convergence rate, theoretically, the best convergence rate for optimizing a convex function using the first-order method has been proven to be $O(\frac{1}{\sqrt{\epsilon}})$ (Nesterov, 2003b). The gap between $O(\frac{1}{\epsilon})$ and $O(\frac{1}{\sqrt{\epsilon}})$ is due to the approximation of the non-separable and non-smooth structured-sparsity-inducing penalty. We can show that if $\mathbf{X}^T\mathbf{X}$ is a positive definite (PD) matrix, $O(\frac{1}{\sqrt{\epsilon}})$ can be achieved by a variant of excessive gap method (Nesterov, 2003a). However, such a rate can not be easily obtained for sparse learning problems where $J > N$ ($\mathbf{X}^T\mathbf{X}$ is not PD). It remains an open question whether we can further boost our algorithm to achieve $O(\frac{1}{\sqrt{\epsilon}})$ convergence rate.

Very recently other works on solving structured sparse learning problems have appeared. For the overlapping-group-lasso penalty, Jenatton et al. (2009) proposed an active-set algorithm that solves a sequence of subproblems with a smaller set of *active* variables. However, this method can only solve the regression problems regularized by the *square* of the structured-sparsity-inducing penalty. In addition, this method formulates each subproblem either as an SOCP, which can be computationally expensive for a large active set, or as a jointly convex problem with auxiliary variables, which is then solved by an alternating gradient descent. This latter approach lacks the guarantee in optimization convergence and may have numerical problems. A variant of proximal gradient method for sparse hierarchical dictionary learning appeared in a very recent work (Jenatton et al., 2010). While their optimization method can be applied only to tree-structured groups, our method can be applied to any arbitrary overlapping group structures.

For the graph-guided fusion penalty, when the structure of the graph is restricted to a chain, pathwise coordinate descent method (Friedman et al., 2007) has been applied. However, this method may not converge to the exact solution when the input matrix \mathbf{X} is not orthogonal. Liu et al. (2010) proposed a first-order method which approximately solves the projection step. However, in this method, the convergence cannot be guaranteed, since the error introduced in each projection step will be accumulated over iterations. For the general graph structure, a different algorithm has been proposed that reformulates the problem as a maximum flow problem (Hoefling, 2009). This algorithm is much more complex than our method, and is limited in that it can be applied only to the case where

the dimension is less than the sample size and that it lacks theoretical guarantees of the convergence.

In addition, we point out that unlike most of the other previous approaches, our method is not specific to any particular type of structured-sparsity-inducing penalties, but it provides a general framework for handling a wide variety of non-separable and non-smooth penalties.

Outside of the optimization community, there exist other works (e.g., Jacob et al. (2009)) that proposed a different type of norms to incorporate the information on group or graph structures. Since the focus of the paper is on the optimization, the comparison of the methods is beyond the scope of this paper.

7. Experiments

In this section, we evaluate the scalability and efficiency of our proximal gradient method (Prox-Grad) as well as the recovery quality of the structured sparsity pattern in regression model on both synthetic and real datasets. For overlapping group lasso, we compare the running time of Prox-Grad with that of SOCP formulation using the standard MATLAB package SDPT3 (Tütüncü et al., 2003). For the graph-guided fused lasso, we compare the running time of Prox-Grad with that of SOCP and QP. For QP formulation, we tried several different packages including MOSEK and CPLEX, and report the results from CPLEX since it performed better than others. All of the experiments are performed on a PC with Intel Core 2 Quad Q6600 2.4GHz CPU and 4GB RAM. The software is written in MATLAB, and we terminate our optimization procedure when the relative change in the objective is below 10^{-6} . We present results for both single-task and multi-task regression problems with various structured-sparsity-inducing penalties. In addition to the comparison of computation time for different optimization algorithms, we present the results on the quality of recovered structured-sparsity patterns.

In our simulation experiment, we constrain the regularization parameters such that $\lambda = \gamma$. We assume that for each group g , $w_g = \sqrt{|g|}$ as described by Yuan and Lin (2006) for the sake of simplicity. We select the regularization parameters using a three-fold cross-validation, and report the computation time as the CPU time for running the optimization procedure on the entire dataset with the selected λ . As for the smoothness parameter μ , following the results in Theorem 10, we set $\mu = \frac{\epsilon}{2D}$, where D is determined by the problem scale. It is natural that for a large-scale problem with a larger D , a larger ϵ can be adopted without affecting the recovery quality significantly. Therefore, instead of fixing the value of ϵ , we directly set $\mu = 10^{-4}$. Our experience showed that this strategy provided us with reasonably good approximation accuracies for different scales of problems.

7.1 Synthetic Data

7.1.1 OVERLAPPING GROUP LASSO FOR SINGLE-TASK REGRESSION

We generate data from known regression coefficients and sparsity structure for a single-task regression, and try to recover the true sparsity pattern using overlapping group lasso. We simulate data using the similar approach as in Jacob et al. (2009), assuming that the inputs have an overlapping group structure as described below. Assuming that the inputs

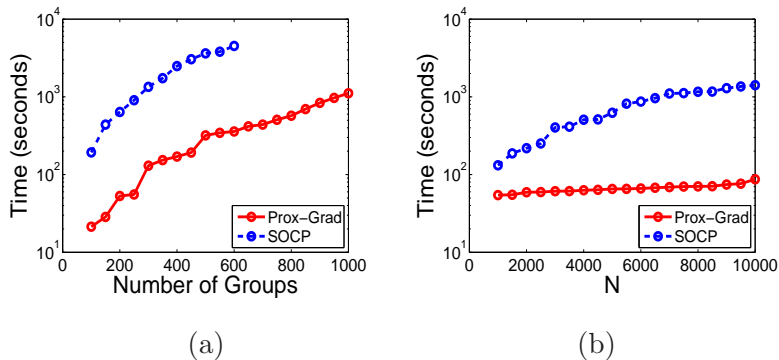


Figure 3: Simulation results for comparing the scalability of Prox-Grad and SOCP with IPM for overlapping group lasso in a single-task regression. (a) Fix $N = 5000$ and vary $|\mathcal{G}|$ from 100 to 1000 with a step size of 50 (i.e., J varies from 703 to 7003). (b) Fix $|\mathcal{G}| = 200$ and vary N from 1000 to 10000 with a step size of 500. The y -axis denotes the computation time in seconds in logarithmic scale.

are ordered, we define a sequence of groups of 10 adjacent inputs with an overlap of three variables between two successive groups so that $\mathcal{G} = \{\{1, \dots, 10\}, \{8, \dots, 17\}, \dots, \{J - 9, \dots, J\}\}$ with $J = 7|\mathcal{G}| + 3$. We set the support of β to the first half of the input variables. We sample each element of \mathbf{X} and the non-zero elements of β from an *i.i.d.* Gaussian distribution, and generate the output data from $\mathbf{y} = \mathbf{X}\beta + \epsilon$, where $\epsilon \sim \mathcal{N}(0, I_{N \times N})$.

To demonstrate the efficiency and scalability of Prox-Grad as compared to SOCP, we present the computation time for datasets with varying N and $|\mathcal{G}|$ in Figure 3. The computation time is measured in seconds and plotted in logarithmic scale. We omit the computation time for the SOCP formulation when it exceeds two hours. Clearly, Prox-Grad is more efficient and scalable by orders of magnitude than IPM for SOCP formulation. Moreover, we notice that the increase of N almost does not affect the computation time of Prox-Grad, which is consistent with our complexity analysis in Section 4.3.

7.1.2 OVERLAPPING GROUP LASSO FOR MULTI-TASK REGRESSION

In this section, we consider the problem of learning a multi-task regression with a overlapping group lasso, where the overlapping group structure is defined over the outputs and related outputs share a similar sparsity pattern in their regression coefficients. We assume that K tasks are organized as a perfect binary tree of depth $l = \log_2(K)$ with leaf nodes corresponding to tasks and internal nodes representing groups of the subtree. Furthermore, we assume that the groups of the tasks near the bottom of the tree are more closely related as in a hierarchical clustering tree and more likely to share the sparsity pattern in their regression coefficients. Given this output structure, we set the sparsity pattern in the true regression-coefficient matrix as shown in Figure 4(a), where the black pixels represent non-zero elements with the value $b = 1.0$ and white pixels represent zero elements. The consecutive black pixels on each column represent a group, and there are many overlaps among these black vertical bars across columns. Then, we sample the elements of \mathbf{X} from

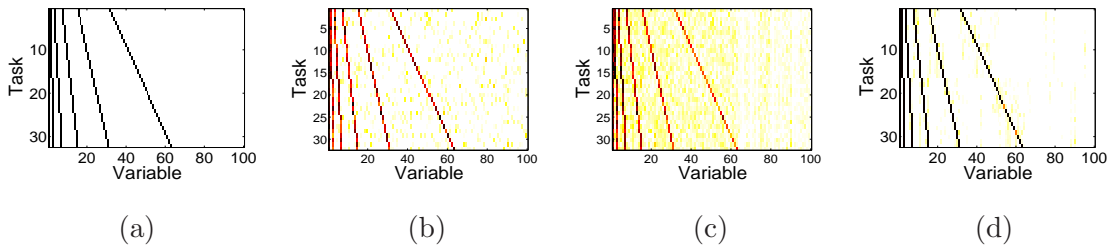


Figure 4: Simulation results for comparing different regression methods on the recovery of tree-structured sparsity pattern in multi-task regression. (a) The matrix of true regression coefficients \mathbf{B} . Estimated regression coefficients are shown for (b) lasso, (c) ℓ_1/ℓ_2 -regularized multi-task lasso, (d) tree-structured overlapping group lasso. The rows and columns represent tasks and inputs, respectively. Black pixels correspond to 1 and white pixels correspond to 0. Red pixels indicate large values approaching to 1 and yellow pixels indicate small values approaching to 0.

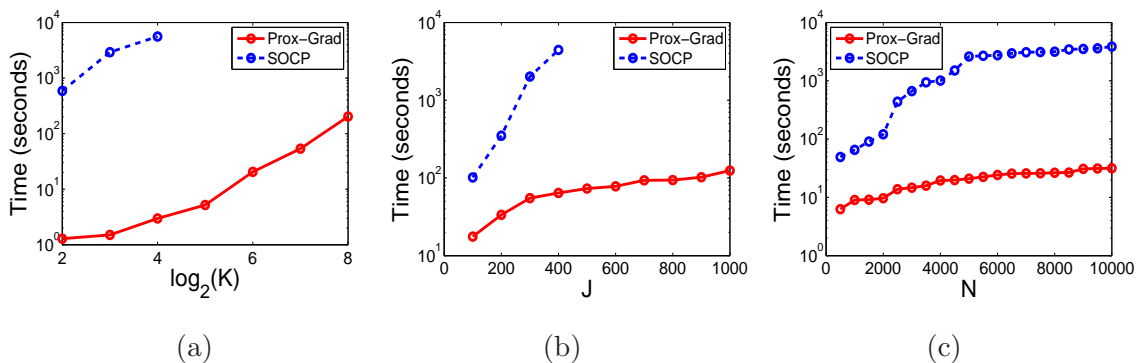


Figure 5: Comparisons of scalability of Prox-Grad and IPM for SOCP using overlapping group lasso in a multi-task regression. (a) Fix $N = 1000$ and $J = 600$, and vary $\log_2(K)$ from 2 to 8 with a step size of 1. (b) Fix $N = 1000$ and $K = 32$, and vary J from 100 to 1000 with a step size of 100. (c) Fix $J = 100$ and $K = 32$, and vary N from 500 to 5000 with a step size of 500. The y -axis shows the computation time in seconds in logarithmic scale.

an *i.i.d.* standard Gaussian and generate the output data using $\mathbf{Y} = \mathbf{X}\mathbf{B} + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon}$ is the standard Gaussian noise.

First, we compare the performance of lasso, the ℓ_1/ℓ_2 -regularized multi-task lasso (Obozinski et al., 2009), and the tree-structured overlapping group lasso in terms of recovery of the true sparsity pattern. We use a dataset simulated with $N = 100$, $J = 100$, and $K = 32$. For the structured multi-task regression, each node in the tree over the outputs defines a group of the tasks. The recovered regression-coefficient matrix is plotted in Figures 4(b)–(d). It is visually clear that the tree-structured overlapping group lasso recovers the true underlying sparsity pattern significantly better than the other methods.

We compare the scalability of the proposed Prox-Grad algorithm with that of IPM for SOCP. We simulate datasets with varying K , J , and N , and present the computational time on these datasets for our Prox-Grad method and IPM for SOCP formulation in Figure

5. For those missing points, SOCP simply cannot be applied since it leads to an out-of-memory error because of the storage for the Newton linear system. As can be seen in Figure 5, our method is significantly faster than the SOCP formulation and can scale up to a very high-dimensional dataset with many tasks.

7.1.3 GRAPH-GUIDED FUSED LASSO FOR MULTI-TASK REGRESSION

In this section, we apply our Prox-Grad method to multi-task graph-guided fused lasso. We simulate data using the following scenario analogous to the problem of genetic association mapping, where we are interested in identifying a small number of genetic variations (inputs) that influence the phenotypes (outputs). We use $K = 10$, $J = 30$ and $N = 100$. To simulate the input data, we use the genotypes of the 60 individuals from the parents of the HapMap CEU panel (The International HapMap Consortium, 2005), and generate genotypes for additional 40 individuals by randomly mating the original 60 individuals. We generate the regression coefficients β_k 's such that the outputs \mathbf{y}_k 's are correlated with a block-like structure in the correlation matrix. We first choose input-output pairs with non-zero regression coefficients as we describe below. We assume three groups of correlated output variables of sizes 3, 3, and 4. We randomly select inputs that are relevant jointly among the outputs within each group, and select additional inputs relevant across multiple groups to model the situation of a higher-level correlation structure across two subgraphs as in Figure 6(a). Given the sparsity pattern of \mathbf{B} , we set all non-zero β_{ij} to a constant $b = 0.8$ to construct the true coefficient matrix \mathbf{B} . Then, we simulate output data based on the linear regression model with noise distributed as standard Gaussian, using the simulated genotypes as inputs. We threshold the output correlation matrix in Figure 6(a) at $\rho = 0.3$ to obtain the graph in Figure 6(b), and use this graph as prior structural information for graph-guided fused lasso.

As an illustrative example, the estimated regression coefficients from different methods are shown in Figures 6(d)–(f). While the results of lasso and ℓ_1/ℓ_2 -regularized multi-task regression in Figures 6(d) and (e) contain many false positives, the results from graph-guided fused lasso in Figure 6(f) show fewer false positives and reveal clear block structures. Thus, graph-guided fused lasso outperforms the other methods.

To compare the scalability of Prox-Grad with those of SOCP and QP, we vary J , N , K , and ρ , and present the computation time in seconds in *logarithmic* scale in Figures 7(a)–(d), respectively. The input data, output data, and true regression coefficients \mathbf{B} are generated in the way similar to what is described above. More precisely, we assume that each group of correlated output variables is of size 10. For each group of the outputs, We randomly select 10% of the input variables as relevant. In addition, we randomly select 5% of the input variables as relevant to every two consecutive groups of outputs and 1% of the input variables as relevant to every three consecutive groups.

In Figure 7, we find that Prox-Grad is substantially more efficient and can scale up to very high-dimensional and large-scale datasets. We point out that when we vary the threshold ρ for generating the output graph in Figure 3(d), the decrease of ρ increases the number of edges $|E|$ and hence increases the computation time. In Figure 3, for large values of J , N , K and small values of ρ , we are unable to collect results for SOCP and QP, because they lead to out-of-memory errors due to the large storage requirement for solving

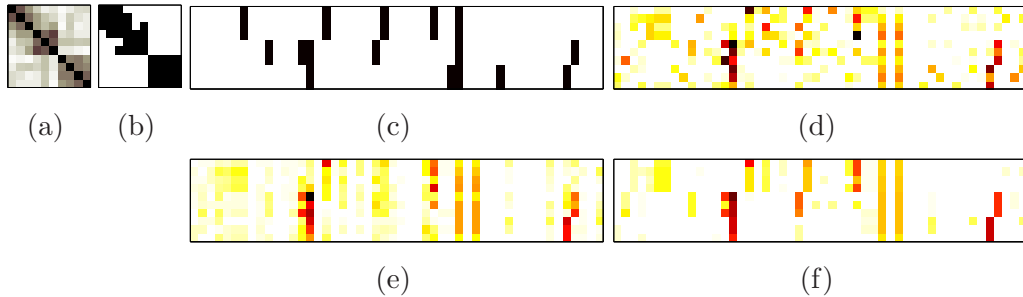


Figure 6: Illustration of the graph-guided fusion penalty for multi-task regression. We show the regression coefficients recovered by different methods based on a single simulated dataset. We use $b = 0.8$ and threshold $\rho = 0.3$ for the output correlation graph. Red pixels indicate large values. (a) The correlation coefficient matrix of outputs, (b) the edges of the phenotype correlation graph obtained at threshold 0.3 are shown as black pixels, (c) the true regression coefficients used in simulation. Absolute values of the estimated regression coefficients are shown for (d) lasso, (e) ℓ_1/ℓ_2 -regularized multi-task lasso, and (f) graph-guided fused lasso. Rows correspond to outputs and columns to inputs.

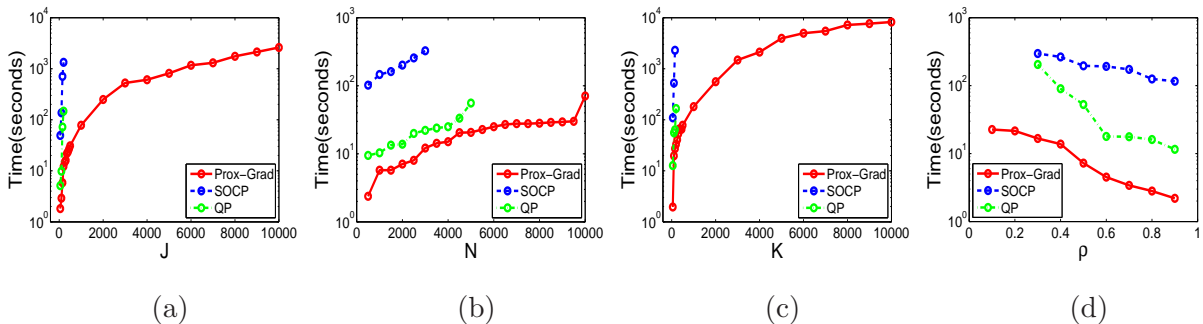


Figure 7: Simulation results for comparing the scalabilities of Prox-Grad, SOCP and QP in a multi-task regression with a graph-guided fusion penalty. (a) Vary J from 50 to 500 with a step size of 50 and then from 1000 to 10,000 with a step size of 1000, fixing $N = 1000$, $K = 50$ and $\rho = 0.5$, (b) Vary N from 500 to 10000 with a step size of 500, fixing $J = 100$, $K = 50$ and $\rho = 0.5$, (c) Vary K from 50 to 500 with a step size of 50 and then from 1000 to 10,000 with a step size of 1000, fixing $N = 500$, $J = 100$ and $\rho = 0.5$, and (d) Vary ρ from 0.1 to 0.9 with a step size of 0.1, fixing $N = 500$, $J = 100$ and $K = 50$. The y -axis shows the computation time in seconds in logarithmic scale.

the Newton linear system. QP is more efficient than SOCP since it removes the non-smooth ℓ_1 penalty by introducing auxiliary variables for each ℓ_1 term. In addition, we notice that the increase of N does not increase the computation time significantly. This observation is consistent with our complexity analysis in Section 4.3.

7.2 Real Data

7.2.1 BREAST CANCER DATA: PATHWAY ANALYSIS

In this section, we apply our Prox-Grad method with the overlapping-group-lasso penalty to a real-world dataset collected from breast cancer tumors (van de Vijver et al., 2002; Jacob et al., 2009) and solve it by proximal gradient method. The data are given as gene expression measurements for 8,141 genes in 296 breast-cancer tumors (78 metastatic and 217 non-metastatic), and the task is to select a small amount of the most relevant genes that gives the best prediction performance.

In a biological system, genes are organized into pathways, and because of the heavy interaction among the genes within a pathway, often the genes in the whole pathway are involved in the development of a disease such as breast cancer. Thus, a powerful way of discovering genes involved in a tumor growth is to consider groups of interacting genes in each pathway rather than individual genes independently (Ma and Kosorok, 2010). The overlapping-group-lasso penalty provides us with a natural way to incorporate these known pathway information into the biological analysis, where each group consists of the genes in each pathway. This approach can allow us to find pathway-level gene groups of significance that can distinguish the two tumor types. In our analysis of the breast cancer data, we cluster the genes using the canonical pathways from the Molecular Signatures Database (Subramanian et al., 2005), and construct the overlapping-group-lasso penalty using the pathway-based clusters as groups. Many of the groups overlap because genes can participate in multiple pathways. Overall, we obtain 637 pathways over 3,510 genes, with each pathway containing 23.47 genes on average and each gene appearing in four pathways on average. Then, we set up the optimization problem of minimizing the logistic loss with the overlapping-group-lasso penalty to classify the tumor types based on the gene expression levels, and solve it with our Prox-Grad method.

Since the number of positive and negative samples are imbalanced, we adopt the balanced error rate defined as the average error rate of the two classes.² We split the data into the training and testing sets with the ratio of 2:1, and vary the $\lambda = \gamma$ from large to small to obtain the full regularization path.

In Figure 8, we compare the results from fitting the logistic regression with the overlapping-group-lasso penalty and the model with the ℓ_1 -norm penalty. Figure 8(a) shows the balanced error rates for the different numbers of selected genes along the regularization path. As we can see, the balanced error rate for the model with the overlapping-group-lasso penalty is lower than the one with ℓ_1 -norm, especially when the number of selected genes is between 500 to 1000. The model with the overlapping-group-lasso penalty achieves the best error rate of 29.23% when 696 genes are selected, and these 696 genes belong to 125 different pathways. In Figure 8(b), for the different numbers of selected genes, we show the number of pathways to which the selected genes belong. From Figure 8(b), we see that when the group structure information is incorporated, fewer pathways are selected. This indicates that learning with the overlapping-group-lasso penalty selects the genes at the pathway level as a functionally coherent groups, leading to an easy interpretation for functional analysis. On the other hand, the genes selected via the ℓ_1 -norm penalty are scattered across many pathways as genes are considered independently for selection. The total computational time

2. See <http://www.modelselect.inf.ethz.ch/evaluation.php> for more details

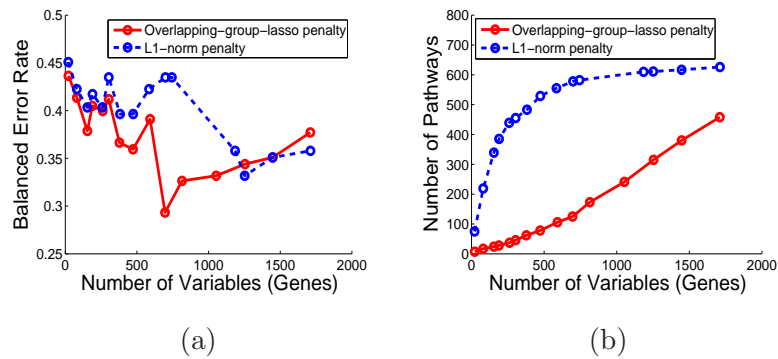


Figure 8: Results from the analysis of breast cancer dataset. (a) Balanced error rate for varying the number of selected genes, and (b) the number of pathways for varying the number of selected genes.

for computing the whole regularization path with 20 different values for the regularization parameters is 331 seconds for the overlapping group lasso.

We perform a functional enrichment analysis on the selected pathways, using the functional annotation tool (Huang et al., 2009), and verify that the selected pathways are significant in their relevance to the breast-cancer tumor types. For example, in a highly sparse model obtained with the group-lasso penalty at the very left end of Figure 8(b), the selected gene markers belong to only seven pathways, and many of these pathways appear to be reasonable candidates for an involvement in breast cancer. For instance, all proteins in one of the selected pathways are involved in the activity of *proteases* whose function is to degrade unnecessary or damaged proteins through a chemical reaction that breaks peptide bonds. One of the most important malignant properties of cancer involves the uncontrolled growth of a group of cells, and *protease inhibitors*, which degrade misfolded proteins, have been extensively studied in the treatment of cancer. Another interesting pathway selected by our method is known for its involvement in *nicotinate and nicotinamide metabolism*. This pathway has been confirmed as a marker for breast cancer in previous studies (Ma and Kosorok, 2010). In particular, the gene *ENPP1* (ectonucleotide pyrophosphatase/phosphodiesterase 1) in this pathway has been found to be overly expressed in breast tumors (Abate et al., 2005). Other selected pathways include the one related to ribosomes and another related to DNA polymerase, which are critical in the process of generating proteins from DNA and relevant to the property of uncontrolled growth in cancer cells.

We also examine the number of selected pathways that gives the lowest error rate in Figure 8. At the error rate of 29.23%, 125 pathways (696 genes) are selected. It is interesting to notice that among these 125 pathways, one is closely related to *apoptosis*, which is the process of programmed cell death that occurs in multicellular organisms and is widely known to be involved in un-controlled tumor growth in cancer. Another pathway involves the genes *BRCA1*, *BRCA2*, and *ATR*, which have all been associated with cancer susceptibility.

For comparison, we examine the genes selected with the ℓ_1 -norm penalty that does not consider the pathway information. In this case, we do not find any meaningful functional enrichment signals that are relevant to breast cancer. For example, among the 582 pathways

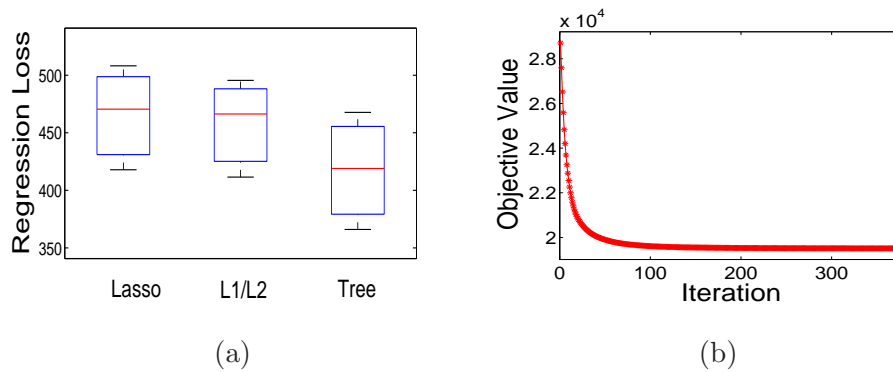


Figure 9: Results on yeast dataset using the overlapping group lasso. (a) Regression error for the test data. (b) The change in values of the objective function over iterations in Prox-Grad.

that involve 687 genes at 37.55% error rate, we find two large pathways with functional enrichments, namely *response to organic substance* (83 genes with p -value $3.3\text{E-}13$) and the process of *oxidation reduction* (73 genes with p -value $1.7\text{E-}11$). However, both are quite large groups and matched to relatively high-level biological processes that do not provide much insight on cancer-specific pathways.

7.2.2 YEAST DATA: GENOME-WIDE ASSOCIATION STUDY

We apply the tree-structured multi-task overlapping group lasso to analyze a genome-wide association study data with 1,260 genotypes (as inputs) and gene expression levels (as outputs) of 3,684 genes collected for 114 yeast strains Zhu et al. (2008). The goal is to select a parsimonious but meaningful set of genotypes that affect gene expressions. We run the hierarchical clustering on the output data to learn the tree structure, where each internal node in the tree defines a group of the tasks (See Kim and Xing (2010) for more details).

We split the data into training and testing sets with the ratio of 2:1, and in Figure 9 (a), report the boxplot of the errors on the test data averaged over five different random splits. Clearly, the structural information leads to a superior performance to lasso and the ℓ_1/ℓ_2 -regularized multi-task lasso.

In Figure 9(b), we show the decrease of the values of the objective function over iterations in a typical run of Prox-Grad for the best selected regularization parameter. Prox-Grad converges in 368 iterations in 1366 seconds. Although the same dataset has been analyzed in (Kim and Xing, 2010), their optimization method based on a variational formulation could handle only a small-scale dataset because it involves an inversion of $J \times J$ matrix in each iteration. Thus, their analysis is focused only on a single chromosome with 21 genotypes instead of the entire set of 1260 genotypes. We note that SOCP simply cannot be applied to this dataset due to the memory issue of storing Newton linear system.

8. Conclusions

In this paper, we considered an optimization problem for learning a structured-sparsity pattern in regression coefficients with a general class of structured-sparsity-inducing penalties. Many of the structured-sparsity-inducing penalties including the overlapping-group-lasso penalties and graph-guided fusion penalty share a common set of difficulties in optimization such as non-separability and non-smoothness. We showed that the optimization problems with these penalties can be transformed into the common form, and proposed a general optimization framework called proximal gradient method that can be applied to an optimization problem of this common form. Our results show that the proposed method can efficiently solve high-dimensional problems.

As future work, we would like to explore other structured penalties that can be efficiently solved by our optimization approach. Another interesting future direction is that since the method is only based on gradient, its online version with the stochastic gradient descent can be easily derived. However, proving the regret bound will require a more careful investigation.

Acknowledgments

We would like to thank Yanjun Qi for the help of preparation and verification of breast cancer dataset. We would also like to thank Javier Peña for the helpful discussion of the related first-order methods. Eric P. Xing is supported by grants ONR N000140910758, NSF DBI-0640543, NSF CCF-0523757, NIH 1R01GM087694, and an Alfred P. Sloan Research Fellowship.

Proof of Theorem 2

The $f_\mu(\beta)$ is a convex function since it is the maximum of a set of functions that are linear in β . For the smoothness property, let the function d^* be the Fenchel conjugate of the strongly-convex function d defined as:

$$d^*(\gamma) = \max_{\alpha \in \mathcal{Q}} \langle \alpha, \gamma \rangle - d(\alpha). \quad (29)$$

We want to prove d^* is differentiable everywhere by showing that the subdifferential ∂d^* of d^* is a singleton set for any γ .

From the definition in (29), for any γ and any $\alpha \in \mathcal{Q}$, we have:

$$d^*(\gamma) + d(\alpha) \geq \langle \alpha, \gamma \rangle, \quad (30)$$

where the inequality holds as an equality if and only if $\alpha = \arg \max_{\alpha' \in \mathcal{Q}} \langle \alpha', \gamma \rangle - d(\alpha')$.

Since d is convex and closed, we have $d^{**} \equiv d$ (Chapter E in Hiriart-Urruty and Lemarechal (2001)). Thus, (30) can be written as:

$$d^*(\gamma) + d^{**}(\alpha) \geq \langle \alpha, \gamma \rangle, \quad (31)$$

where the inequality holds as an equality if and only if $\gamma = \arg \max_{\gamma' \in \mathbb{R}^J} \langle \alpha, \gamma' \rangle - d^*(\gamma')$.

Since (30) and (31) are equivalent, we know that $\alpha = \arg \max_{\alpha' \in \mathcal{Q}} \alpha'^T \gamma - d(\alpha')$ if and only if $\gamma = \arg \max_{\gamma' \in \mathbb{R}^J} \langle \alpha, \gamma' \rangle - d^*(\gamma')$. The latter equality implies that for any γ' :

$$d^*(\gamma') \geq d^*(\gamma) + \langle \alpha, \gamma' - \gamma \rangle,$$

which further means that α is a subgradient of d^* at γ by the definition of subgradient.

Summarizing the above arguments, we conclude that α is a subgradient of d^* at γ if and only if

$$\alpha = \arg \max_{\alpha' \in \mathcal{Q}} \langle \alpha', \gamma \rangle - d(\alpha'). \quad (32)$$

Since d is a strongly-convex function, this maximization problem in (32) has a unique optimal solution. Thus, the subdifferential ∂d^* of d^* at any point γ is a singleton set that contains only α . Therefore, d^* is differentiable everywhere (Chapter D in Hiriart-Urruty and Lemarechal (2001)) and α is its gradient:

$$\nabla d^*(\gamma) = \alpha = \arg \max_{\alpha' \in \mathcal{Q}} \langle \alpha', \gamma \rangle - d(\alpha'). \quad (33)$$

Now we return to our original problem of $f_\mu(\beta)$ and rewrite it as:

$$f_\mu(\beta) = \max_{\alpha \in \mathcal{Q}} \langle \alpha, C\beta \rangle - \mu d(\alpha) = \mu \max_{\alpha \in \mathcal{Q}} \left[\langle \alpha, \frac{C\beta}{\mu} \rangle - d(\alpha) \right] = \mu d^*\left(\frac{C\beta}{\mu}\right).$$

Using (33) and the chain rule, we know that $f_\mu(\beta)$ is continuously differentiable and its gradient takes the following form:

$$\begin{aligned} \nabla f_\mu(\beta) &= \mu C^T (\nabla d^*\left(\frac{C\beta}{\mu}\right)) = \mu C^T \left(\arg \max_{\alpha' \in \mathcal{Q}} \left[\langle \alpha', \frac{C\beta}{\mu} \rangle - d(\alpha') \right] \right) \\ &= C^T \left(\arg \max_{\alpha' \in \mathcal{Q}} [\langle \alpha', C\beta \rangle - \mu d(\alpha')] \right) = C^T(\alpha^*). \end{aligned}$$

For the proof of Lipschitz constant of $f_\mu(\beta)$, readers can refer to Nesterov (2005).

Proof of Proposition 4

Since we have

$$\|C\mathbf{v}\|_2 = \gamma \sqrt{\sum_{g \in \mathcal{G}} \sum_{j \in g} (w_g)^2 v_j^2} = \lambda \sqrt{\sum_{j=1}^J \left(\sum_{g \in \mathcal{G} \text{ s.t. } j \in g} (w_g)^2 \right) v_j^2},$$

the maximum value of $\|C\mathbf{v}\|_2$, given $\|\mathbf{v}\|_2 \leq 1$, can be achieved by setting $v_{\hat{j}}$ for \hat{j} corresponding to the largest summation $\sum_{g \in \mathcal{G} \text{ s.t. } j \in g} (w_g)^2$ to one, and setting other v_j 's to zeros. Hence, we have $\|C\mathbf{v}\|_2 = \gamma \max_{j \in \{1, \dots, J\}} \sqrt{\sum_{g \in \mathcal{G} \text{ s.t. } j \in g} (w_g)^2}$.

Proof of Proposition 6

According to the construction of matrix C , we have for any vector \mathbf{v} :

$$\|C\mathbf{v}\|_2^2 = \gamma^2 \sum_{e=(m,l) \in E} (\tau(r_{ml}))^2 (v_m - \text{sign}(r_{ml})v_l)^2. \quad (34)$$

By the simple fact that $(a \pm b)^2 \leq 2a^2 + 2b^2$ and the inequality holds as equality if and only if $a = \pm b$, for each edge $e = (m, l) \in E$, the value $(v_m - \text{sign}(r_{ml})v_l)^2$ is upper-bounded by $2v_m^2 + 2v_l^2$. Hence, when $\|\mathbf{v}\|_2 = 1$, the right-hand side of (34) can be further bounded by:

$$\begin{aligned} \|C\mathbf{v}\|_2^2 &\leq \gamma^2 \sum_{e=(m,l) \in E} 2(\tau(r_{ml}))^2 (v_m^2 + v_l^2) \\ &= \gamma^2 \sum_{j \in V} (\sum_{e \text{ incident on } j} 2(\tau(r_e))^2) v_j^2 \\ &= \gamma^2 \sum_{j \in V} 2d_j v_j^2 \\ &\leq 2\gamma^2 \max_{j \in V} d_j, \end{aligned} \quad (35)$$

where d_j is defined in (14). Therefore, we have

$$\|C\| \equiv \max_{\|\mathbf{v}\|_2 \leq 1} \|C\mathbf{v}\|_2 \leq \sqrt{2\gamma^2 \max_{j \in V} d_j}.$$

Note that this upper bound is tight because the first inequality in (35) is tight.

Proof of Theorem 10

Based on the result from Beck and Teboulle (2009), we have the following lemma:

Lemma 14 *For the function $\tilde{f}(\boldsymbol{\beta}) = h(\boldsymbol{\beta}) + \lambda\|\boldsymbol{\beta}\|_1$, where $h(\boldsymbol{\beta})$ is an arbitrary convex smooth function and its gradient $\nabla h(\boldsymbol{\beta})$ is Lipschitz continuous with the Lipschitz constant L . We apply Algorithm 1 to minimize $\tilde{f}(\boldsymbol{\beta})$ and let $\boldsymbol{\beta}^t$ be the approximate solution at the t -th iteration. For any $\boldsymbol{\beta}$, we have the following bound:*

$$\tilde{f}(\boldsymbol{\beta}^t) - \tilde{f}(\boldsymbol{\beta}) \leq \frac{2L\|\boldsymbol{\beta} - \boldsymbol{\beta}^0\|_2^2}{t^2}. \quad (36)$$

In order to use the bound in (36), we decompose $f(\boldsymbol{\beta}^t) - f(\boldsymbol{\beta}^*)$ into three terms:

$$f(\boldsymbol{\beta}^t) - f(\boldsymbol{\beta}^*) = \left(f(\boldsymbol{\beta}^t) - \tilde{f}(\boldsymbol{\beta}^t) \right) + \left(\tilde{f}(\boldsymbol{\beta}^t) - \tilde{f}(\boldsymbol{\beta}^*) \right) + \left(\tilde{f}(\boldsymbol{\beta}^*) - f(\boldsymbol{\beta}^*) \right). \quad (37)$$

According to the definition of \tilde{f} , we know that for any $\boldsymbol{\beta}$

$$\tilde{f}(\boldsymbol{\beta}) \leq f(\boldsymbol{\beta}) \leq \tilde{f}(\boldsymbol{\beta}) + \mu D,$$

where $D \equiv \max_{\boldsymbol{\alpha} \in \mathcal{Q}} d(\boldsymbol{\alpha})$. Therefore, the first term in (37), $f(\boldsymbol{\beta}^t) - \tilde{f}(\boldsymbol{\beta}^t)$, is upper-bounded by μD , and the last term in (37) is less than or equal to 0 (i.e., $\tilde{f}(\boldsymbol{\beta}^*) - f(\boldsymbol{\beta}^*) \leq 0$). Combining (36) with these two simple bounds, we have:

$$f(\boldsymbol{\beta}^t) - f(\boldsymbol{\beta}^*) \leq \mu D + \frac{2L\|\boldsymbol{\beta}^* - \boldsymbol{\beta}^0\|_2^2}{t^2} \leq \mu D + \frac{2\|\boldsymbol{\beta}^* - \boldsymbol{\beta}^0\|_2^2}{t^2} \left(\lambda_{\max}(\mathbf{X}^T \mathbf{X}) + \frac{\|C\|^2}{\mu} \right). \quad (38)$$

By setting $\mu = \frac{\epsilon}{2D}$ and plugging this into the right-hand side of (38), we obtain

$$f(\boldsymbol{\beta}^t) - f(\boldsymbol{\beta}^*) \leq \frac{\epsilon}{2} + \frac{2\|\boldsymbol{\beta}^* - \boldsymbol{\beta}^0\|_2^2}{t^2} \left(\lambda_{\max}(\mathbf{X}^T \mathbf{X}) + \frac{2D\|C\|^2}{\epsilon} \right). \quad (39)$$

If we require the right-hand side of (39) to be equal to ϵ and solve it for t , we obtain the bound of t in (21).

Note that we can set $\mu = \frac{\epsilon}{h}$ for any $h > 1$ to achieve $O\left(\frac{1}{\epsilon}\right)$ convergence rate, which is different from (21) only by a constant factor.

Proof of Proposition 13

Given any $\mathbf{x} \in \mathbb{R}^J$ and $\boldsymbol{\beta}, \boldsymbol{\alpha} \in \mathbb{R}^J$, we have

$$\begin{aligned} & \left| \frac{e^{\boldsymbol{\beta}^T \mathbf{x}}}{1 + e^{\boldsymbol{\beta}^T \mathbf{x}}} - \frac{e^{\boldsymbol{\alpha}^T \mathbf{x}}}{1 + e^{\boldsymbol{\alpha}^T \mathbf{x}}} \right| \\ & \leq \frac{|e^{\boldsymbol{\beta}^T \mathbf{x}} + e^{\boldsymbol{\beta}^T \mathbf{x} + \boldsymbol{\alpha}^T \mathbf{x}} - e^{\boldsymbol{\alpha}^T \mathbf{x}} - e^{\boldsymbol{\alpha}^T \mathbf{x} + \boldsymbol{\beta}^T \mathbf{x}}|}{(1 + e^{\boldsymbol{\beta}^T \mathbf{x}})(1 + e^{\boldsymbol{\alpha}^T \mathbf{x}})} \\ & \leq \frac{|e^{\boldsymbol{\beta}^T \mathbf{x}} - e^{\boldsymbol{\alpha}^T \mathbf{x}}|}{(1 + e^{\boldsymbol{\beta}^T \mathbf{x}})(1 + e^{\boldsymbol{\alpha}^T \mathbf{x}})} \\ & \leq |\boldsymbol{\beta}^T \mathbf{x} - \boldsymbol{\alpha}^T \mathbf{x}| \\ & \leq \|\boldsymbol{\beta} - \boldsymbol{\alpha}\|_2 \|\mathbf{x}\|_2. \end{aligned}$$

According to the above inequality, we can derive the following inequalities:

$$\begin{aligned}
& \|\nabla f(\boldsymbol{\beta}) - \nabla f(\boldsymbol{\alpha})\|_2 \\
= & \left\| \sum_{i=1}^N \mathbf{x}_i \left(\frac{e^{\boldsymbol{\beta}^T \mathbf{x}_i}}{1 + e^{\boldsymbol{\beta}^T \mathbf{x}_i}} - \frac{e^{\boldsymbol{\alpha}^T \mathbf{x}_i}}{1 + e^{\boldsymbol{\alpha}^T \mathbf{x}_i}} \right) \right\|_2 \\
\leq & \sum_{i=1}^N \left\| \mathbf{x}_i \left(\frac{e^{\boldsymbol{\beta}^T \mathbf{x}_i}}{1 + e^{\boldsymbol{\beta}^T \mathbf{x}_i}} - \frac{e^{\boldsymbol{\alpha}^T \mathbf{x}_i}}{1 + e^{\boldsymbol{\alpha}^T \mathbf{x}_i}} \right) \right\|_2 \\
\leq & \sum_{i=1}^N \|\mathbf{x}_i\|_2^2 \|\boldsymbol{\beta} - \boldsymbol{\alpha}\|_2.
\end{aligned}$$

References

- Nicola Abate, Manisha Chandalia, Pankaj Satija, Beverley Adams-Huet, and et. al. Enpp1/pc-1 k121q polymorphism and genetic susceptibility to type 2 diabetes. *Diabetes*, 54(4):1027–1213, 2005.
- Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Convex multi-task feature learning. *Machine Learning*, 73:243–272, 2008.
- Amir Beck and Marc Teboulle. A fast iterative shrinkage thresholding algorithm for linear inverse problems. *SIAM Journal of Image Science*, 2(1):183–202, 2009.
- John Duchi and Yoram Singer. Efficient online and batch learning using forward backward splitting. *Journal of Machine Learning Research*, 10:2899–2934, 2009.
- Jerome Friedman, Trevor Hastie, Holger Höfling, and Robert Tibshirani. Pathwise coordinate optimization. *Annals of Applied Statistics*, 1:302–332, 2007.
- Jean-Baptiste Hiriart-Urruty and Claude Lemarechal. *Fundamentals of Convex Analysis*. Springer, 2001.
- Holger Hoefling. A path algorithm for the fused lasso signal approximator. arXiv:0910.0526v1 [stat.CO], 2009.
- Da Wei Huang, Brad T Sherman, and Richard A Lempicki. Systematic and integrative analysis of large gene lists using david bioinformatics resources. *Nature Protoc*, 4(1): 44–57, 2009.
- Laurent Jacob, Guillaume Obozinski, and Jean-Philippe Vert. Group lasso with overlap and graph lasso. In *Proceedings of the 26th International Conference on Machine Learning*, 2009.
- Rodolphe Jenatton, Jean-Yves Audibert, and Francis Bach. Structured variable selection with sparsity-inducing norms. Technical report, INRIA, 2009.

- Rodolphe Jenatton, Julien Mairal, Guillaume Obozinski, and Francis Bach. Proximal methods for sparse hierarchical dictionary learning. In *Proceedings of the 27th International Conference on Machine Learning*, 2010.
- Seyoung Kim and Eric P. Xing. Tree-guided group lasso for multi-task regression with structured sparsity. In *Proceedings of the 27th International Conference on Machine Learning*, 2010.
- Seyoung Kim, Kyung-Ah Sohn, and Eric P. Xing. A multivariate regression approach to association analysis of a quantitative trait network. *Bioinformatics*, 25(12):204–212, 2009.
- Jun Liu, Shuiwang Ji, and Jieping Ye. Multi-task feature learning via efficient $\ell_{2,1}$ -norm minimization. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2009.
- Jun Liu, Lei Yuan, and Jieping Ye. An efficient algorithm for a class of fused lasso problems. In *the 16th ACM SIGKDD*, 2010.
- Miguel Sousa Lobo, Lieven Vandenberghe, Stephen Boyd, and Herve Lebret. Applications of second-order cone programming. *Linear Algebra and Its Applications*, 284:193–228, 1998.
- Shuangge Ma and Michael R Kosorok. Detection of gene pathways with predictive power for breast cancer prognosis. *BMC Bioinformatics*, 11(1), 2010.
- Yurii Nesterov. Excessive gap technique in non-smooth convex minimization. Technical report, Universit catholique de Louvain, Center for Operations Research and Econometrics (CORE), 2003a.
- Yurii Nesterov. *Introductory lectures on convex optimization: a basic course*. Kluwer Academic Pub, 2003b.
- Yurii Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103(1):127–152, 2005.
- Yurri Nesterov. Gradient methods for minimizing composite objective function. Technical report, Universit catholique de Louvain, Center for Operations Research and Econometrics (CORE), 2007.
- Guillaume Obozinski, Ben Taskar, and Michael I. Jordan. High-dimensional union support recovery in multivariate regression. In *Advances in Neural Information Processing Systems (NIPS)*. MIT Press, 2009.
- A. Subramanian, P. Tamayo, V. Mootha, and et. al. Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences*, 102(43):15545–15550, 2005.
- The International HapMap Consortium. A haplotype map of the human genome. *Nature*, 437:1399–1320, 2005.
- Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B*, 58:267–288, 1996.

- Robert Tibshirani and Michael Saunders. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society, Series B*, 67(1):91–108, 2005.
- Reha H. Tütüncü, Kim C. Toh, and Michael J. Todd. Solving semidefinite-quadratic-linear programs using sdpt3. *Mathematical Programming*, 95:189–217, 2003.
- Marc J. van de Vijver et al. A gene-expression signature as a predictor of survival in breast cancer. *New England Journal of Medicine*, 347:1999–2009, 2002.
- Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B*, 68:49–67, 2006.
- Peng Zhao, Guilherme Rocha, and Bin Yu. The composite absolute penalties family for grouped and hierarchical variable selection. *The Annals of Statistics*, 37(6A):3468–3497, 2009.
- Jun Zhu, Bin Zhang, Erin Smith, and et. al. Integrating large-scale functional genomic data to dissect the complexity of yeast regulatory networks. *Nature Genetics*, 40:854–861, 2008.