Submitted to the Annals of Applied Statistics

SMOOTHING PROXIMAL GRADIENT METHOD FOR GENERAL STRUCTURED SPARSE REGRESSION

By Xi Chen , Qihang Lin, Seyoung Kim , Jaime G. Carbonell and Eric P. Xing*

Carnegie Mellon University

We study the problem of estimating high dimensional regression models regularized by a structured sparsity-inducing penalty that encodes prior structural information on either the input or output variables. We consider two widely adopted types of penalties of this kind as motivating examples: 1) the general overlapping-group-lasso penalty, generalized from the group-lasso penalty; and 2) the graph-guided-fused-lasso penalty, generalized from the fused-lasso penalty. For both types of penalties, due to their non-separability and non-smoothness, developing an efficient optimization method remains a challenging problem. In this paper, we propose a general optimization approach, the *smoothing proximal gradient* (SPG) method, which can solve structured sparse regression problems with any smooth convex loss under a wide spectrum of structured sparsity-inducing penalties. Our approach combines a smoothing technique with effective proximal gradient method. It achieves a convergence rate significantly faster than the standard first-order methods, subgradient methods, and is much more scalable than the most widely used interior-point methods. The efficiency and scalability of our method are demonstrated on both simulation experiments and real genetic datasets.

1. Introduction. The problem of high-dimensional sparse feature learning arises in many areas in science and engineering. In a typical setting such as linear regression, the input signal leading to a response (i.e., the output) lies in a high-dimensional space, and one is interested in selecting a small number of truly relevant variables in the input 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,

*Supported by Grants ONR N000140910758, NSF DBI-0640543, NSF CCF-0523757, NIH 1R01GM087694, AFOSR FA95501010247, NIH 1R01GM093156 and an Alfred P. Sloan Research Fellowship awarded to EPX. Correspondence should be addressed to Eric P. Xing.

AMS 2000 subject classifications: Primary 90C25; secondary 90C06

Keywords and phrases: Sparse Regression, Structured Sparsity, Smoothing, Proximal Gradient, Optimization

this approach is limited in that it is incapable of capturing any structural information among the input variables. Recently, various extensions of the ℓ_1-norm lasso penalty have been introduced to take advantage of the prior knowledge of the structures 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 multivariate-response regression (or multi-task regression), where one is interested in estimating multiple related functional mappings from a common input space to multiple outputs (Obozinski *et al.*, 2009; Kim and Xing, 2009, 2010). 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. These progresses notwithstanding, the development of efficient optimization methods for solving the estimation problems resultant from the *structured sparsity-inducing penalty functions* remains a challenge for reasons we will discuss bellow. In this paper, we address the problem of developing efficient optimization methods that can handle a broad family of structured sparsity-inducing penalties with complex structures.

When the structure to be imposed during shrinkage has a relatively simple form, such as non-overlapping groups over variables (e.g., group lasso (Yuan and Lin, 2006)), or a linear-ordering (a.k.a., chain) of variables (e.g., fused lasso (Tibshirani and Saunders, 2005)), efficient optimization methods have been developed. For example, under group lasso, due to the separability among groups, a *proximal operator*[1] associated with the penalty can be computed in closed-form; thus, a number of composite gradient methods (Beck and Teboulle, 2009; Nesterov, 2007; Liu *et al.*, 2009) that leverage the proximal operator as a key step (so-called "proximal gradient method") can be directly applied. For fused lasso, although the penalty is not separable, a coordinate descent algorithm was shown feasible by explicitly leveraging the linear ordering of the inputs (Friedman *et al.*, 2007).

Unfortunately, these algorithmic advancements have been outpaced by the emergence of more complex structures one would like to impose during shrinkage. For example, in order to handle a more general class of structures such as a tree or a graph over variables, various regression models that further extend the group lasso and fused lasso ideas have been recently proposed. Specifically, rather than assuming the variable groups to be non-overlapping as in the standard group lasso, the *overlapping group lasso* (Jenatton *et al.*, 2009) allows each input variable to belong to multi-

[1]The proximal operator associated with the penalty is defined as: $\arg\min_{\boldsymbol{\beta}} \frac{1}{2}\|\boldsymbol{\beta}-\mathbf{v}\|_2^2 + P(\boldsymbol{\beta})$, where $\mathbf{v}$ is any given vector and $P(\boldsymbol{\beta})$ is the non-smooth penalty.

ple groups, thereby introducing overlaps among groups and enabling incorporation of more complex prior knowledge on the structure. Going beyond the standard fused lasso, the *graph-guided fused lasso* extends the original chain structure over variables to a general graph over variables, where the fused-lasso penalty is applied to each edge of the graph (Kim *et al.*, 2009). Due to the non-separability of the penalty terms resultant from the overlapping group or graph structures in these new models, the aforementioned fast optimization methods originally tailored for the standard group lasso or fused lasso cannot be readily applied here, due to, for example, unavailability of a closed-form solution of the proximal operator. In principle, generic convex optimization solvers such as the interior-point methods (IPM) could always be used to solve either a second-order cone programming (SOCP) or a quadratic programming (QP) formulation of the aforementioned problems; but such approaches are computationally prohibitive for problems of even a moderate size. Very recently, a great deal of attentions have been given to devise practical solutions to the complex structured sparse regression problems discussed above in statistics and machine learning community, and numerous methods have been proposed (Jenatton *et al.*, 2010; Mairal *et al.*, 2010; Duchi and Singer, 2009; Liu *et al.*, 2010; Zhou and Lange, 2011; Tibshirani and Taylor, 2010). All of these recent works strived to provide clever solutions to various subclasses of the structured sparsity-inducing penalties; but, as we survey in Section 4, they are still short of reaching a simple, unified, and general solution to a broad class of structured sparse regression problems.

In this paper, we propose a generic optimization approach, the *smoothing proximal gradient* (SPG) method, for dealing with a broad family of sparsity-inducing penalties of complex structures. We use the overlapping-group-lasso penalty and graph-guided-fused-lasso penalty mentioned above as our motivating examples. Although these two types of penalties are seemingly very different, we show that it is possible to decouple the non-separable terms in both penalties via the dual norm; and reformulate them into a common form to which the proposed method can be applied. We call our approach a "smoothing" proximal gradient method because instead of optimizing the original objective function directly as in other proximal gradient methods, we introduce a *smooth* approximation to the structured sparsity-inducing penalty using the technique from Nesterov (2005). Then, we solve the smoothed surrogate problem by a first-order proximal gradient method known as the fast iterative shrinkage-thresholding algorithm (FISTA)(Beck and Teboulle, 2009). We show that although we solve a smoothed problem, when the smoothness parameter is carefully chosen, SPG achieves a convergence rate of $O(\frac{1}{\epsilon})$ for the original objective for any

desired accuracy ϵ. Below, we summarize the main advantages of this approach:

(a) It is a first-order method, as it uses only the gradient information. Thus, it is significantly more scalable than IPM for SOCP or QP. Since it is gradient-based, it allows warm restarts, thereby potentiates solving the problem along the entire regularization path (Friedman *et al.*, 2007).
(b) It is applicable to a wide class of optimization problems with a smooth convex loss and a non-smooth non-separable structured sparsity-inducing penalty. Additionally, it is applicable to both uni- and multi-task sparse structured regression, with structures on either (or both) inputs/outputs.
(c) Theoretically, it enjoys a convergence rate of $O(\frac{1}{\epsilon})$, which dominates that of the standard first-order method such as subgradient method whose rate is of $O(\frac{1}{\epsilon^2})$.
(d) Finally, SPG is easy to implement with a few lines of MATLAB code.

The idea of constructing a smoothing approximation to a difficult-to-optimize objective function has also been adopted in another widely used optimization framework known as majorization-minimization (MM) (Lange, 2004). Using the quadratic surrogate functions for the ℓ_2-norm and fused-lasso penalty as derived in Wu and Lange (2008) and Zhang *et al.* (2010), one can also apply MM to solve the structured sparse regression problems. We will discuss in detail the connections between our methods and MM in Section 4.

The rest of this paper is organized as follows. In Section 2, we present the formulation of overlapping group lasso and graph-guided fused lasso. In Section 3, we present the SPG method along with complexity results. In Section 4, we discuss the connections between our method and MM, and comparisons with other related methods. In Section 5, we extend our algorithm to multivariate-task regression. In Section 6, we present numerical results on both simulated and real datasets, followed by conclusions in Section 7. Throughout the paper, we will discuss overlapping-group-lasso and graph-guided-fused-lasso penalties in parallel to illustrate how the SPG can be used to solve the corresponding optimization problems generically.

2. Background: Linear Regression Regularized by Structured Sparsity-inducing Penalties. We begin with a basic outline of the high-dimensional linear regression model, regularized by structured sparsity-inducing penalties.

Consider a data set of N feature/response (i.e., input/output) pairs, $\{\mathbf{x}_n, y_n\}$, $n = 1, \ldots, N$. Let $\mathbf{X} \in \mathbb{R}^{N \times J}$ denote the matrix of inputs of the N samples, where each sample lies in a J-dimensional space; and $\mathbf{y} \in \mathbb{R}^{N \times 1}$

denote the vector of uni-variate outputs of the N sample. Under a linear regression model, $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$, where $\boldsymbol{\beta}$ represents the vector of length J for the regression coefficients, and $\boldsymbol{\epsilon}$ is the vector of length N for noise distributed as $N(0, \sigma^2 I_{N\times N})$. The well known Lasso regression (Tibshirani, 1996) obtains a sparse estimate of the coefficients by solving the following optimization problem:

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^J} g(\boldsymbol{\beta}) + \lambda\|\boldsymbol{\beta}\|_1, \tag{2.1}$$

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.

The standard lasso penalty does not assume any structure among the input variables, which limits its applicability to complex high-dimensional scenarios in many applied problems. More structured constraints on the input variables such as groupness or pairwise similarities can be introduced by employing a more sophisticated sparsity-inducing penalty that induces joint sparsity patterns among related inputs. We generically denote the structured sparsity-inducing penalty by $\Omega(\boldsymbol{\beta})$ without assuming a specific form, and define the problem of estimating a structured sparsity pattern of the coefficients as follows:

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^J} f(\boldsymbol{\beta}) \equiv g(\boldsymbol{\beta}) + \Omega(\boldsymbol{\beta}) + \lambda\|\boldsymbol{\beta}\|_1. \tag{2.2}$$

In this paper, we consider two types of $\Omega(\boldsymbol{\beta})$ that capture two different kinds of structural constraints over variables, namely, the overlapping-group-lasso penalty based on the ℓ_1/ℓ_2 mixed-norm, and the graph-guided-fused-lasso penalty based on a total variation norm. As we discuss below, these two types of penalties represent a broad family of structured sparsity-inducing penalties recently introduced in the literature (Yuan and Lin, 2006; Jenatton *et al.*, 2009; Kim and Xing, 2010; Zhao *et al.*, 2009a; Tibshirani and Saunders, 2005; Kim *et al.*, 2009). It is noteworthy that in problem (2.2), in addition to the structured-sparsity-inducing penalty $\Omega(\boldsymbol{\beta})$, there is also an ℓ_1-regularizer $\lambda\|\boldsymbol{\beta}\|_1$ that explicitly enforces sparsity on every individual features. The SPG optimization algorithm to be presented in this paper is applicable regardless of the presence or absence of the $\lambda\|\boldsymbol{\beta}\|_1$ term.

1. **Overlapping-group-lasso penalty**
   Given prior knowledge of (possibly overlapping) grouping of variables or features, if it is desirable to encourage coefficients of features within the same group to be shrunk to zero jointly, then a composite structured penalty of the following form can be used:

$$\Omega(\boldsymbol{\beta}) \equiv \gamma \sum_{g \in \mathcal{G}} w_g \|\boldsymbol{\beta}_g\|_2, \tag{2.3}$$

where $\mathcal{G} = \{g_1, \ldots, g_{|\mathcal{G}|}\}$ denotes the set of groups, which is a subset of the power set of $\{1, \ldots, J\}$; $\boldsymbol{\beta}_g \in \mathbb{R}^{|g|}$ is the subvector of $\boldsymbol{\beta}$ for the features in group g; w_g is the predefined weight for group g; and $\|\cdot\|_2$ is the vector ℓ_2-norm. This ℓ_1/ℓ_2 mixed-norm penalty plays the role of jointly setting all of the coefficients within each group to zero or non-zero values. The widely used hierarchical tree-structured penalty (Zhao *et al.*, 2009b; Kim and Xing, 2010) is a special case of (2.3), of which the groups are defined as a nested set under a tree hierarchy. It is noteworthy that the ℓ_1/ℓ_∞ mixed-norm penalty can also achieve a similar grouping effect. Indeed our approach can also be applied to the ℓ_1/ℓ_∞ penalty, but for simplicity here we focus on only the ℓ_1/ℓ_2 penalty and the comparison between the ℓ_1/ℓ_2 and the ℓ_1/ℓ_∞ is beyond the scope of the paper.

Apparently, the penalty $\Omega(\boldsymbol{\beta}) \equiv \gamma \sum_{g \in \mathcal{G}} w_g \|\boldsymbol{\beta}_g\|_2$ alone enforces only group-level sparsity but not sparsity within each group. More precisely, if the estimated $\|\widehat{\boldsymbol{\beta}}_g\|_2 \neq 0$, each $\widehat{\beta}_j$ for $j \in g$ will be non-zero. By using an additional ℓ_1-regularizer $\lambda\|\boldsymbol{\beta}\|_1$ together with $\Omega(\boldsymbol{\beta})$ as in (2.2), one can not only select groups but also variables within each group. The readers may refer to Friedman *et al.* (2010) for more details.

2. **Graph-guided-fused-lasso penalty**
Alternatively, prior knowledge about the structural constraints over features can be in the form of their pairwise relatedness described by a graph $G \equiv (V, E)$, where $V = \{1, \ldots, J\}$ denotes the variables or features of interest, and E denotes the set of edges among V. Additionally, we let $r_{ml} \in \mathbb{R}$ denote the weight of the edge $e = (m, l) \in E$, corresponding to correlation or other proper similarity measures between features m and l. If it is desirable to encourage coefficients of related features to share similar magnitude, then the graph-guided-fused-lasso penalty (Kim *et al.*, 2009) of the following form can be used:

$$\Omega(\boldsymbol{\beta}) = \gamma \sum_{e=(m,l) \in E, m<l} \tau(r_{ml}) |\beta_m - \operatorname{sign}(r_{ml}) \beta_l|, \tag{2.4}$$

where $\tau(r_{ml})$ represent a general weight function that enforces a fusion effect over coefficients β_m and β_l of relevant features. In this paper, we

consider $\tau(r) = |r|$, but any monotonically increasing function of the absolute values of correlations can be used.
The $\text{sign}(r_{ml})$ in (2.4) ensures that two positively correlated inputs would tend to influence the output in the same direction, whereas two negatively correlated inputs impose opposite effect. Since the fusion effect is calibrated by the edge weight, the graph-guided-fused-lasso penalty in (2.4) encourages highly inter-correlated inputs corresponding to a densely connected subnetwork in G to be jointly selected as relevant.
It is noteworthy that when $r_{ml} = 1$ for all $e = (m, l) \in E$, and G is simply a chain over nodes, we have:

$$\Omega(\boldsymbol{\beta}) = \gamma \sum_{j=1}^{J-1} |\beta_{j+1} - \beta_j|, \tag{2.5}$$

which is identical to the standard fused lasso penalty (Tibshirani and Saunders, 2005).

3. Smoothing Proximal Gradient. Although (2.2) defines a convex program, of which globally optimal solution to $\boldsymbol{\beta}$ is attainable, the main difficulty in solving (2.2) arises from the non-separability of elements of $\boldsymbol{\beta}$ in the non-smooth penalty function $\Omega(\boldsymbol{\beta})$. As we show in the next subsection, although the overlapping-group-lasso and graph-guided-fused-lasso penalties are seemingly very different, we can reformulate the two types of penalties as a common matrix algebraic form, to which a generic Nesterov smoothing technique can be applied. The key 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. Based on that, we introduce a smooth approximation to $\Omega(\boldsymbol{\beta})$ using the technique from Nesterov (2005) such that its gradient with respect to $\boldsymbol{\beta}$ can be easily calculated.

3.1. *Reformulation of Structured Sparsity-inducing Penalty.* In this section, we show that utilizing the dual norm, the non-separable structured sparsity-inducing penalty in both (2.3) and (2.4) can be decoupled; and reformulated into a common form as a maximization problem over the auxiliary variables.

1. **Reformulating overlapping-group-lasso penalty**
Since the dual norm of an ℓ_2-norm is also ℓ_2-norm, we can write $\|\boldsymbol{\beta}_g\|_2$ as $\|\boldsymbol{\beta}_g\|_2 = \max_{\|\boldsymbol{\alpha}_g\|_2 \leq 1} \boldsymbol{\alpha}_g^T \boldsymbol{\beta}_g$, where $\boldsymbol{\alpha}_g \in \mathbb{R}^{|g|}$ is a vector of auxiliary variables associated with $\boldsymbol{\beta}_g$. Let $\boldsymbol{\alpha} = \left[\boldsymbol{\alpha}_{g_1}^T, \ldots, \boldsymbol{\alpha}_{g_{|\mathcal{G}|}}^T\right]^T$. Then, $\boldsymbol{\alpha}$ is a

vector of length $\sum_{g\in G}|g|$ with domain $\mathcal{Q}\equiv\{\boldsymbol{\alpha}\mid\|\boldsymbol{\alpha}_g\|_2\leq 1,\ \forall g\in\mathcal{G}\}$, where $\mathcal{Q}$ is the Cartesian product of unit balls in Euclidean space and therefore, a closed and convex set. We can rewrite the overlapping-group-lasso penalty in (2.3) as:

(3.1)
$$\Omega(\boldsymbol{\beta})=\gamma\sum_{g\in\mathcal{G}}w_g\max_{\|\boldsymbol{\alpha}_g\|_2\leq 1}\boldsymbol{\alpha}_g^T\boldsymbol{\beta}_g=\max_{\boldsymbol{\alpha}\in\mathcal{Q}}\sum_{g\in\mathcal{G}}\gamma w_g\boldsymbol{\alpha}_g^T\boldsymbol{\beta}_g=\max_{\boldsymbol{\alpha}\in\mathcal{Q}}\boldsymbol{\alpha}^TC\boldsymbol{\beta},$$

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 $(i,g)\in\{(i,g)|i\in g,i\in\{1,\ldots,J\},g\in\mathcal{G}\}$, the columns are indexed by $j\in\{1,\ldots,J\}$, and each element of C is given as:

$$C_{(i,g),j}=\begin{cases}\gamma w_g & \text{if } i=j,\\ 0 & \text{otherwise.}\end{cases}\tag{3.2}$$

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.

2. **Reformulating graph-guided-fused-lasso penalty**
First, we rewrite the graph-guided-fused-lasso penalty in (2.4) as follows:
$$\gamma\sum_{e=(m,l)\in E,m<l}\tau(r_{ml})|\beta_m-\text{sign}(r_{ml})\beta_l|\equiv\|C\boldsymbol{\beta}\|_1,$$
where $C\in\mathbb{R}^{|E|\times J}$ is the edge-vertex incident matrix:
$$C_{e=(m,l),j}=\begin{cases}\gamma\cdot\tau(r_{ml}) & \text{if } j=m\\ -\gamma\cdot\text{sign}(r_{ml})\tau(r_{ml}) & \text{if } j=l\\ 0 & \text{otherwise.}\end{cases}\tag{3.3}$$
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-fused-lasso penalty as:
$$\|C\boldsymbol{\beta}\|_1\equiv\max_{\|\boldsymbol{\alpha}\|_\infty\leq 1}\boldsymbol{\alpha}^TC\boldsymbol{\beta},\tag{3.4}$$
where $\boldsymbol{\alpha}\in\mathcal{Q}=\{\boldsymbol{\alpha}|\|\boldsymbol{\alpha}\|_\infty\leq 1,\boldsymbol{\alpha}\in\mathbb{R}^{|E|}\}$ 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. *As a generalization of graph-guided-fused-lasso penalty, the proposed optimization method can be applied to the ℓ_1-norm of any linear mapping of $\boldsymbol{\beta}$ (i.e., $\Omega(\boldsymbol{\beta})=\|C\boldsymbol{\beta}\|_1$ for any given C).*

3.2. *Smooth Approximation to Structured Sparsity-inducing Penalty.* The common formulation of $\Omega(\boldsymbol{\beta})$ given above (i.e., $\Omega(\boldsymbol{\beta}) = \max_{\boldsymbol{\alpha}\in\mathcal{Q}} \boldsymbol{\alpha}^T C\boldsymbol{\beta}$) is still a non-smooth function of $\boldsymbol{\beta}$, and this makes the optimization challenging. To tackle this problem, using the technique from Nesterov (2005), we construct a smooth approximation to $\Omega(\boldsymbol{\beta})$ as following:

$$f_\mu(\boldsymbol{\beta}) = \max_{\boldsymbol{\alpha}\in\mathcal{Q}} \left(\boldsymbol{\alpha}^T C\boldsymbol{\beta} - \mu d(\boldsymbol{\alpha})\right), \tag{3.5}$$

where μ is a positive smoothness parameter and $d(\boldsymbol{\alpha})$ is a smoothing function defined as $\frac{1}{2}\|\boldsymbol{\alpha}\|_2^2$. The original penalty term can be viewed as $f_\mu(\boldsymbol{\beta})$ with $\mu = 0$; and one can verify 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 $D = \max_{\boldsymbol{\alpha}\in\mathcal{Q}} d(\boldsymbol{\alpha})$. In our problems, $D = |\mathcal{G}|/2$ for the overlapping-group-lasso penalty and $D = |E|/2$ for the graph-guided-fused-lasso penalty. Then, it is easy to verify that the maximum gap between $f_\mu(\boldsymbol{\beta})$ and $f_0(\boldsymbol{\beta})$ is μD:

$$f_0(\boldsymbol{\beta}) - \mu D \leq f_\mu(\boldsymbol{\beta}) \leq f_0(\boldsymbol{\beta}).$$

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* to $f_0(\boldsymbol{\beta})$ with a maximum gap of μD; and the μ controls the gap between $f_\mu(\boldsymbol{\beta})$ and $f_0(\boldsymbol{\beta})$. Given a desired accuracy ϵ, the convergence result in Section 3.5 suggests $\mu = \frac{\epsilon}{2D}$ to achieve the best convergence rate.

Now we present the key theorem (Nesterov, 2005) to show that $f_\mu(\boldsymbol{\beta})$ is smooth in $\boldsymbol{\beta}$ with a simple form of the gradient.

THEOREM 1. *For any $\mu > 0$, $f_\mu(\boldsymbol{\beta})$ is a convex and continuously-differentiable function in $\boldsymbol{\beta}$, and the gradient of $f_\mu(\boldsymbol{\beta})$ takes the following form:*

$$\nabla f_\mu(\boldsymbol{\beta}) = C^T\boldsymbol{\alpha}^*, \tag{3.6}$$

where $\boldsymbol{\alpha}^*$ is the optimal solution to (3.5). Moreover, the gradient $\nabla f_\mu(\boldsymbol{\beta})$ is Lipschitz continuous with the Lipschitz constant $L_\mu = \frac{1}{\mu}\|C\|^2$, where $\|C\|$ is the matrix spectral norm of C defined as $\|C\| \equiv \max_{\|\mathbf{v}\|_2\leq 1} \|C\mathbf{v}\|_2$.

By viewing $f_\mu(\boldsymbol{\beta})$ as the *Fenchel Conjugate* of $d(\cdot)$ at $\frac{C\boldsymbol{\beta}}{\mu}$, the smoothness can be obtained by applying Theorem 26.3 in Rockafellar (1996). The gradient in (3.6) can be derived from the Danskin's Theorem (Bertsekas, 1999) and the Lipschitz constant is shown in Nesterov (2005). The details of the proof are given in the appendix.

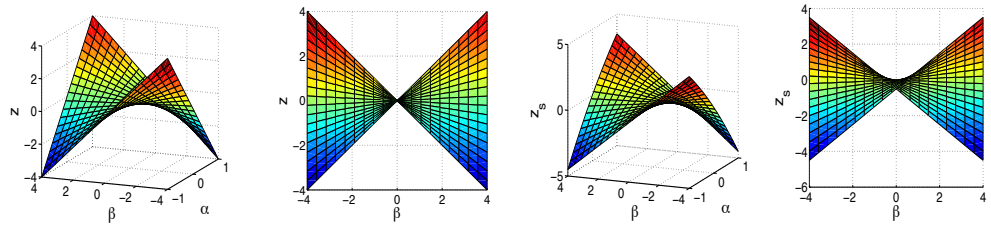


FIG 1. *A geometric illustration of the smoothness of $f_\mu(\beta)$. (a) The 3-D plot of $z(\alpha, \beta)$, (b) the projection of (a) onto the β-z space, (c) the 3-D plot of $z_s(\alpha, \beta)$, and (d) the projection of (c) onto the β-z space.*

Geometric illustration of Theorem 1 To provide insights on why $f_\mu(\boldsymbol{\beta})$ 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., $\beta \in \mathbb{R}$) with μ and C set to 1. First, we show geometrically that $f_0(\beta) = \max_{\alpha\in[-1,1]} z(\alpha, \beta)$ with $z(\alpha, \beta) \equiv \alpha\beta$ 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 $\beta - z$ 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 $\beta = 0$ and hence is *non-smooth*. Now, we introduce $d(\alpha) = \frac{1}{2}\alpha^2$, 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 $\beta - z_s$ 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 the sharp point at $\beta = 0$ is removed and $f_\mu(\beta)$ becomes *smooth*.

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-fused-lasso penalty in the following propositions. The proof is presented in the appendix.

1. **$\boldsymbol{\alpha}^*$ under overlapping-group-lasso penalty**

PROPOSITION 1. *Let $\boldsymbol{\alpha}^*$, which is composed of $\{\boldsymbol{\alpha}_g^*\}_{g\in\mathcal{G}}$, be the optimal solution to (3.5) for the overlapping-group-lasso penalty in (2.3). For any $g \in \mathcal{G}$,*

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

where S is the projection operator which projects any vector $\mathbf{u}$ to the ℓ_2 ball:

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

In addition, we have $\|C\| = \gamma \max_{j\in\{1,\ldots,J\}} \sqrt{\sum_{g\in\mathcal{G} \ s.t. \ j\in g} (w_g)^2}$.

2. **$\boldsymbol{\alpha}^*$ under graph-guided-fused-lasso penalty**

PROPOSITION 2. *Let $\boldsymbol{\alpha}^*$ be the optimal solution of (3.5) for graph-guided-fused-lasso penalty in (2.4). Then, we have:*

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

where S is the projection operator defined as follows:

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

For any vector $\boldsymbol{\alpha}$, $S(\boldsymbol{\alpha})$ is defined as applying S on each and every entry of $\boldsymbol{\alpha}$.
$\|C\|$ is upper-bounded by $\sqrt{2\gamma^2 \max_{j\in V} d_j}$, where

$$d_j = \sum_{e\in E \ s.t. \ e \text{ incident on } j} (\tau(r_e))^2 \tag{3.7}$$

for $j \in V$ in graph G, and this bound is tight. Note that when $\tau(r_e) = 1$ for all $e \in E$, d_j is simply the degree of the node j.

3.3. *Smoothing Proximal Gradient Descent.* Given the smooth approximation to the non-smooth structured sparsity-inducing penalties, now, we apply the fast iterative shrinkage-thresholding algorithm (FISTA) (Beck and Teboulle, 2009; Tseng, 2008) to solve a generically reformulated optimization problem, using the gradient information from Theorem 1. We substitute the penalty term $\Omega(\boldsymbol{\beta})$ in (2.2) with its smooth approximation $f_\mu(\boldsymbol{\beta})$ to obtain the following optimization problem:

$$\min_{\boldsymbol{\beta}} \widetilde{f}(\boldsymbol{\beta}) \equiv g(\boldsymbol{\beta}) + f_\mu(\boldsymbol{\beta}) + \lambda\|\boldsymbol{\beta}\|_1. \tag{3.8}$$

Algorithm 1 Smoothing Proximal Gradient Descent (SPG) for Structured Sparse Regression

Input: $\mathbf{X}$, $\mathbf{y}$, C, $\boldsymbol{\beta}^0$, Lipschitz constant L, desired accuracy ϵ.
Initialization: set $\mu = \frac{\epsilon}{2D}$ where $D = \max_{\boldsymbol{\alpha}\in\mathcal{Q}} \frac{1}{2}\|\boldsymbol{\alpha}\|_2^2$ ($D = |\mathcal{G}|/2$ for the overlapping-group-lasso penalty and $D = |E|/2$ for the graph-guided-fused-lasso penalty), $\theta_0 = 1$, $\mathbf{w}^0 = \boldsymbol{\beta}^0$.
Iterate For $t = 0, 1, 2, \ldots$, until convergence of $\boldsymbol{\beta}^t$:

1. Compute $\nabla h(\mathbf{w}^t)$ according to (3.10).
2. Solve the proximal operator associated with the ℓ_1-norm:
   (3.12)
   $$\boldsymbol{\beta}^{t+1} = \arg\min_{\boldsymbol{\beta}} Q_L(\boldsymbol{\beta}, \mathbf{w}^t) \equiv h(\mathbf{w}^t) + \langle \boldsymbol{\beta} - \mathbf{w}^t, \nabla h(\mathbf{w}^t)\rangle + \lambda\|\boldsymbol{\beta}\|_1 + \frac{L}{2}\|\boldsymbol{\beta} - \mathbf{w}^t\|_2^2$$
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: $\widehat{\boldsymbol{\beta}} = \boldsymbol{\beta}^{t+1}$.

Let

$$h(\boldsymbol{\beta}) = g(\boldsymbol{\beta}) + f_\mu(\boldsymbol{\beta}) = \frac{1}{2}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + f_\mu(\boldsymbol{\beta}). \tag{3.9}$$

be the smooth part of $\widetilde{f}(\boldsymbol{\beta})$. According to Theorem 1, the gradient of $h(\boldsymbol{\beta})$ is given as:

$$\nabla h(\boldsymbol{\beta}) = \mathbf{X}^T(\mathbf{X}\boldsymbol{\beta} - \mathbf{y}) + C^T\boldsymbol{\alpha}^*. \tag{3.10}$$

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

$$L = \lambda_{\max}(\mathbf{X}^T\mathbf{X}) + L_\mu = \lambda_{\max}(\mathbf{X}^T\mathbf{X}) + \frac{\|C\|^2}{\mu}, \tag{3.11}$$

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

Since $\widetilde{f}(\boldsymbol{\beta})$ 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 $\widetilde{f}(\boldsymbol{\beta})$ as shown in Algorithm 1. Algorithm 1 alternates between the sequences $\{w^t\}$ and $\{\boldsymbol{\beta}^t\}$ and θ_t can be viewed as a special "step-size", which determines the relationship between $\{w^t\}$ and $\{\boldsymbol{\beta}^t\}$ as in Step 4 of Algorithm 1. As shown in Beck and Teboulle (2009), such a way of setting θ_t leads to Lemma 1 in Appendix, which further guarantees the convergence result in Theorem 2.

Rewriting $Q_L(\boldsymbol{\beta}, \mathbf{w}^t)$ in (3.12):

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

Let $\mathbf{v} = (\mathbf{w}^t - \frac{1}{L}\nabla h(\mathbf{w}^t))$, the closed-form solution for $\boldsymbol{\beta}^{t+1}$ can be obtained by soft-thresholding (Friedman *et al.*, 2007) as presented in the next proposition.

PROPOSITION 3. *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:

$$\beta_j = \text{sign}(v_j)\max(0, |v_j| - \frac{\lambda}{L}), \quad j = 1, \ldots, J. \tag{3.13}$$

An important advantage of using the proximal operator associated with the ℓ_1-norm $Q_L(\boldsymbol{\beta}, \mathbf{w}^t)$ is that it can provide us with sparse solutions, where the coefficients for irrelevant inputs are set *exactly* to zeros, due to the soft-thresholding operation in (3.13). When the term $\lambda\|\boldsymbol{\beta}\|_1$ is not included in the objective, for overlapping group lasso, we can only obtain the group level sparsity but not the individual feature level sparsity inside each group. However, as for optimization, Algorithm 1 still applies in the same way. The only difference is that Step 2 of Algorithm 1 becomes $\boldsymbol{\beta}^{t+1} = \arg\min_{\boldsymbol{\beta}} h(\mathbf{w}^t) + \langle \boldsymbol{\beta} - \mathbf{w}^t, \nabla h(\mathbf{w}^t)\rangle + \frac{L}{2}\|\boldsymbol{\beta} - \mathbf{w}^t\|_2^2 = \mathbf{w}^t - \frac{1}{L}\nabla h(\mathbf{w}^t)$. Since there is no soft-thresholding step, the obtained solution $\widehat{\boldsymbol{\beta}}$ has no exact zeros. We then need to set a threshold (e.g., 10^{-5}) and select the relevant groups which contain the variables with the parameter above this threshold.

3.4. *Issues on the Computation of the Lipschitz Constant.* When J is large, the computation of $\lambda_{\max}(\mathbf{X}^T\mathbf{X})$ and hence the Lipschitz constant L could be very expensive. To further accelerate Algorithm 1, a line search backtracking step could be used to dynamically assign a constant L_t for the proximal operator in each iteration (Beck and Teboulle, 2009). More specifically, given any positive constant R, let

$$Q_R(\boldsymbol{\beta}, \mathbf{w}^t) = h(\mathbf{w}^t) + \langle \boldsymbol{\beta} - \mathbf{w}^t, \nabla h(\mathbf{w}^t)\rangle + \lambda\|\boldsymbol{\beta}\|_1 + \frac{R}{2}\|\boldsymbol{\beta} - \mathbf{w}^t\|_2^2,$$

and

$$\boldsymbol{\beta}^{t+1} \equiv \boldsymbol{\beta}_R(\mathbf{w}^t) = \arg\min_{\boldsymbol{\beta}} Q_R(\boldsymbol{\beta}, \mathbf{w}^t).$$

The key to guarantee the convergence rate of Algorithm 1 is to ensure that the following inequality holds for each iteration:

$$\widetilde{f}(\boldsymbol{\beta}^{t+1}) = h(\boldsymbol{\beta}^{t+1}) + \lambda\|\boldsymbol{\beta}^{t+1}\|_1 \leq Q_R(\boldsymbol{\beta}^{t+1}, \mathbf{w}^t). \tag{3.14}$$

It is easy to check when R is equal to the Lipschitz constant L, it will satisfy the above inequality for any $\boldsymbol{\beta}^{t+1}$ and $\mathbf{w}^t$. However, when it is difficult to compute the Lipschitz constant, instead of using a global constant L, we could find a sequence $\{L_t\}_{t=0}^T$ such that L_{t+1} satisfies the inequality (3.14) for the t-th iteration. In particular, we start with any small constant L_0. For each iteration, we find the smallest integer $a \in \{0, 1, 2, \ldots\}$ such that by setting $L_{t+1} = \tau^a L_t$ where $\tau > 1$ is a pre-defined scaling factor, we have:

$$\widetilde{f}(\boldsymbol{\beta}_{L_{t+1}}(\mathbf{w}^t)) \leq Q_{L_{t+1}}(\boldsymbol{\beta}_{L_{t+1}}(\mathbf{w}^t), \mathbf{w}^t). \tag{3.15}$$

Then we set $\boldsymbol{\beta}^{t+1} = \boldsymbol{\beta}_{L_{t+1}}(\mathbf{w}^t) \equiv \arg\min Q_{L_{t+1}}(\boldsymbol{\beta}, \mathbf{w}^t)$.

3.5. *Convergence Rate and Time Complexity.* Although we optimize the approximation function $\widetilde{f}(\boldsymbol{\beta})$ rather than the original $f(\boldsymbol{\beta})$ directly, it can be proven that $f(\widehat{\boldsymbol{\beta}})$ is sufficiently close to the optimal objective value of the *original* function $f(\boldsymbol{\beta}^*)$. The convergence rate of Algorithm 1 is presented in the next theorem.

THEOREM 2. *Let $\boldsymbol{\beta}^*$ be the optimal solution to (2.2) and $\boldsymbol{\beta}^t$ be the approximate solution at the t-th iteration in Algorithm 1. If we require $f(\boldsymbol{\beta}^t) - f(\boldsymbol{\beta}^*) \leq \epsilon$ where f is the original objective, and set $\mu = \frac{\epsilon}{2D}$, then the number of iterations t is upper-bounded by*

$$\sqrt{\frac{4\|\boldsymbol{\beta}^* - \boldsymbol{\beta}^0\|_2^2}{\epsilon}\left(\lambda_{\max}(\mathbf{X}^T\mathbf{X}) + \frac{2D\|C\|^2}{\epsilon}\right)}. \tag{3.16}$$

The key idea behind the proof of this theorem is to decompose $f(\boldsymbol{\beta}^t) - f(\boldsymbol{\beta}^*)$ into three parts: (i) $f(\boldsymbol{\beta}^t) - \widetilde{f}(\boldsymbol{\beta}^t)$, (ii) $\widetilde{f}(\boldsymbol{\beta}^t) - \widetilde{f}(\boldsymbol{\beta}^*)$, and (iii) $\widetilde{f}(\boldsymbol{\beta}^*) - f(\boldsymbol{\beta}^*)$. (i) and (iii) can be bounded by the gap of the approximation μD; and (ii) only involves the function $\widetilde{f}$ and can be upper bounded by $O(\frac{1}{t^2})$ as shown in Beck and Teboulle (2009). We obtain (3.16) by balancing these three terms. The details of the proof are presented in the appendix. According to Theorem 2, 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.

REMARK 2. *Since there is no line search in Algorithm 1, we cannot guarantee that the objective values are monotonically decreasing over iterations theoretically. But empirically, based on our own experience, the objective values always decrease over iterations. One simple strategy to guarantee the*

TABLE 1
Comparison of Per-iteration Time Complexity

	Overlapping Group Lasso	Graph-guided Fused Lasso
SPG	$O(J\min(J,N)+\sum_{g\in\mathcal{G}}\lvert g\rvert)$	$O(J\min(J,N)+\lvert E\rvert)$
IPM	$O\left((J+\lvert\mathcal{G}\rvert)^2(N+\sum_{g\in\mathcal{G}}\lvert g\rvert)\right)$	$O\left((J+\lvert E\rvert)^3\right)$

monotone decreasing property is to first compute $\widetilde{\boldsymbol{\beta}}^{t+1} = arg\,min_{\boldsymbol{\beta}} Q_L(\boldsymbol{\beta}, \mathbf{w}^t)$ *and then set* $\boldsymbol{\beta}^{t+1} = arg\,min_{\boldsymbol{\beta}\in\{\widetilde{\boldsymbol{\beta}}^{t+1},\boldsymbol{\beta}^t\}} f(\boldsymbol{\beta})$.

REMARK 3. *Theorem 2 only shows the convergence rate for the objective value. As for the estimator* $\boldsymbol{\beta}^t$*, since it is a convex optimization problem, it is well known that* $\boldsymbol{\beta}^t$ *will eventually converge to* $\boldsymbol{\beta}^*$*. However, the speed of convergence of* $\boldsymbol{\beta}^t$ *to* $\boldsymbol{\beta}^*$ *depends on the structure of the input* $\mathbf{X}$*. If* $h(\boldsymbol{\beta})$ *is a strongly convex function with the strongly convexity parameter* $\sigma > 0$*. In our problem, it is equivalent to saying that* $\mathbf{X}^T\mathbf{X}$ *is a non-singular matrix with the smallest eigenvalue* $\sigma > 0$*. Then we can show that if* $f(\boldsymbol{\beta}^t) - f(\boldsymbol{\beta}^*) \leq \epsilon$ *at the convergence, then* $\|\boldsymbol{\beta}^t - \boldsymbol{\beta}^*\|_2 \leq \sqrt{\frac{2\epsilon}{\sigma}}$*. In other words,* $\boldsymbol{\beta}^t$ *converges to* $\boldsymbol{\beta}^*$ *in* ℓ_2*-distance at the rate of* $O(\frac{1}{\epsilon^2})$*. For general high-dimensional sparse learning problems with* $J > N$*,* $\mathbf{X}^T\mathbf{X}$ *is singular and hence the optimal solution* $\boldsymbol{\beta}^*$ *is not unique. In such a case, one can only show that* $\boldsymbol{\beta}^t$ *will converge to one of the optimal solutions. But the speed of the convergence of* $\|\boldsymbol{\beta}^t - \boldsymbol{\beta}^*\|_2$ *or its relationship with* $f(\boldsymbol{\beta}^t) - f(\boldsymbol{\beta}^*)$ *is widely recognized as an open problem in optimization community.*

As for the time complexity, the main computational cost in each iteration comes from calculating the gradient $\nabla h(\mathbf{w}_t)$. Therefore, SPG shares almost the same per-iteration time as the subgradient descent but with a faster convergence rate. In more details, if $J < N$ and $\mathbf{X}^T\mathbf{X}$ and $\mathbf{X}^T\mathbf{y}$ can be pre-computed and stored in memory, the computation of first part of $\nabla h(\mathbf{w}_t)$, $(\mathbf{X}^T\mathbf{X})\mathbf{w}_t - (\mathbf{X}^T\mathbf{y})$, takes the time complexity of $O(J^2)$. Otherwise, if $J > N$, we can compute this part by $\mathbf{X}^T(\mathbf{X}\mathbf{w}_t - \mathbf{y})$ which takes the time complexity of $O(JN)$. As for the generic solver, IPM for SOCP for overlapping group lasso or IPM for QP for graph-guided fused lasso, although it converges in fewer iterations (i.e., $\log(\frac{1}{\epsilon})$), its per-iteration complexity is higher by orders of magnitude than ours as shown in Table 1. In addition to time complexity, IPM requires the pre-storage of $\mathbf{X}^T\mathbf{X}$ and each IPM iteration requires significantly more memory to store the Newton linear system. Therefore, the SPG is much more efficient and scalable for large-scale problems.

3.6. *Summary and Discussions.* The insight of our work was drawn from two lines of earlier works. The first one is the proximal gradient methods (e.g., Nesterov's composite gradient method (Nesterov, 2007), FISTA (Beck and Teboulle, 2009)). They have been widely adopted to solve optimization problems with a convex loss and a relatively simple non-smooth penalty, achieving $O(\frac{1}{\sqrt{\epsilon}})$ convergence rate. However, the complex structure of the non-separable penalties considered in this paper makes it intractable to solve the proximal operator exactly. This is the challenge that we circumvent via smoothing.

The general idea of the smoothing technique used in this paper was first introduced by Nesterov (2005). The algorithm presented in Nesterov (2005) only works for smooth problems so that it has to smooth out the entire non-smooth penalty. Our approach separates the simple non-smooth ℓ_1-norm penalty from the complex structured sparsity-inducing penalties. In particular, when an ℓ_1-norm penalty is used to enforce the *individual-feature-level sparsity* (which is especially necessary for fused lasso), we smooth out the complex structured-sparsity-inducing penalty while leaving the simple ℓ_1-norm as it is. One benefit of our approach is that it can lead to solutions with *exact zeros* for irrelevant features due to the ℓ_1-norm penalty and hence avoid the post-processing (i.e., truncation) step[2]. Moreover, the algorithm in Nesterov (2005) requires the condition that $\boldsymbol{\beta}$ is bounded and that the number of iterations is pre-defined, which are impractical for real applications.

As for the convergence rate, the gap between $O\left(\frac{1}{\epsilon}\right)$ and the optimal rate $O(\frac{1}{\sqrt{\epsilon}})$ is due to the approximation of the structured sparsity-inducing penalty. It is possible to show that if $\mathbf{X}$ has a full column rank, $O(\frac{1}{\sqrt{\epsilon}})$ can be achieved by a variant of excessive gap method (Nesterov, 2003). However, such a rate cannot be easily obtained for sparse regression problems where $J > N$. For some special cases as discussed in the next Section, such as tree-structured or the ℓ_1/ℓ_∞ mixed-norm based overlapping groups, $O(\frac{1}{\sqrt{\epsilon}})$ can be achieved at the expense of more computation time for solving the proximal operator. It remains an open question whether we can further boost the generally-applicable SPG method to achieve $O(\frac{1}{\sqrt{\epsilon}})$.

[2]When there is no ℓ_1-norm penalty in the model (i.e., $\lambda = 0$), our method still applies. However, to conduct variable selection, as for other optimization methods (e.g., IPM), we need a post-processing step to truncate parameters below a certain threshold to zeros.

4. Related Optimization Methods.

4.1. *Connections with Majorization-Minimization.* The idea of constructing a smoothing approximation has also been adopted in another widely used optimization method, majorization-minimization (MM) for minimization problem (or minorization-maximization for maximization problem) (Lange, 2004). To minimize a given objective, MM replaces the difficult-to-optimize objective function with a simple (and smooth in most cases) surrogate function which majorizes the objective. It minimizes the surrogate function and iterates such a procedure. The difference between our approach and MM is that our approximation is a uniformly smooth lower bound of the objective with a bounded gap; whereas the surrogate function in MM is an upper bound of the objective. In addition, MM is an iterative procedure which iteratively constructs and minimizes the surrogate function; while our approach constructs the smooth approximation once and then applies the proximal gradient descent to optimize it. With the quadratic surrogate functions for the ℓ_2-norm and fused-lasso penalty derived in Wu and Lange (2008) and Zhang *et al.* (2010), one can easily apply MM to solve the structured sparse regression problems. However, in our problems, the Hessian matrix in the quadratic surrogate will no longer have a simple structure (e.g. tridiagonal symmetric structure in chain-structured fused signal approximator). Therefore, one may need to apply the general optimization methods, e.g., conjugate-gradient or quasi-Newton method, to solve a series of quadratic surrogate functions. In addition, since the objective functions considered in our paper are neither smooth nor strictly convex, the local and global convergence results for MM in Lange (2004) cannot be applied. It seems to us still an open problem to derive the local, global convergence and the convergence rate for MM for the general non-smooth convex optimization.

Recently, many first-order approaches have been developed for various subclasses of overlapping group lasso and graph-guided fused lasso. Below, we provide a survey of these methods:

4.2. *Related work for mixed-norm based group-lasso penalty.* Most of the existing optimization methods developed for mixed-norm penalties can handle only a specific subclass of the general overlapping-group-lasso penalties. Most of these methods use the proximal gradient framework (Beck and Teboulle, 2009; Nesterov, 2007) and focus on the issue of how to *exactly* solve the proximal operator. For non-overlapping groups with the ℓ_1/ℓ_2 or ℓ_1/ℓ_∞ mixed-norms, the proximal operator can be solved via a simple projection (Liu *et al.*, 2009; Duchi and Singer, 2009). A one-pass coordinate ascent method has been developed for tree-structured groups with the ℓ_1/ℓ_2

TABLE 2
Comparisons of different first-order methods for optimizing mixed-norm based overlapping-group-lasso penalties.

Method	No overlap ℓ_1/ℓ_2	No overlap ℓ_1/ℓ_∞	Overlap Tree ℓ_1/ℓ_2	Overlap Tree ℓ_1/ℓ_∞	Overlap Arbitrary ℓ_1/ℓ_2	Overlap Arbitrary ℓ_1/ℓ_∞
Projection (Liu *et al.*, 2009)	$O(\frac{1}{\sqrt{\epsilon}})$, $O(J)$	$O(\frac{1}{\sqrt{\epsilon}})$, $O(J \log J)$	N.A.	N.A.	N.A.	N.A.
Coordinate Ascent (Jenatton *et al.*, 2010; Liu and Ye, 2010b)	$O(\frac{1}{\sqrt{\epsilon}})$, $O(J)$	$O(\frac{1}{\sqrt{\epsilon}})$, $O(J \log J)$	$O(\frac{1}{\sqrt{\epsilon}})$, $O(\sum_{g\in\mathcal{G}} \lvert g\rvert)$	$O(\frac{1}{\sqrt{\epsilon}})$, $O(\sum_{g\in\mathcal{G}} \lvert g\rvert \log \lvert g\rvert)$	N.A.	N.A.
Network Flow (Mairal *et al.*, 2010)	N.A.	$O(\frac{1}{\sqrt{\epsilon}})$, quadratic min-cost flow	N.A.	$O(\frac{1}{\sqrt{\epsilon}})$, quadratic min-cost flow	N.A.	$O(\frac{1}{\sqrt{\epsilon}})$, quadratic min-cost flow
FOBOS (Duchi and Singer, 2009)	$O(\frac{1}{\epsilon})$, $O(J)$	$O(\frac{1}{\epsilon})$, $O(J \log J)$	$O(\frac{1}{\epsilon})$, $O(\sum_{g\in\mathcal{G}} \lvert g\rvert)$	$O(\frac{1}{\epsilon})$, $O(\sum_{g\in\mathcal{G}} \lvert g\rvert \log \lvert g\rvert)$	$O(\frac{1}{\epsilon^2})$, $O(\sum_{g\in\mathcal{G}} \lvert g\rvert)$ (subgradient)	$O(\frac{1}{\epsilon})$, quadratic min-cost flow
SPG	$O(\frac{1}{\epsilon})$, $O(J)$	$O(\frac{1}{\epsilon})$, $O(J \log J)$	$O(\frac{1}{\epsilon})$, $O(\sum_{g\in\mathcal{G}} \lvert g\rvert)$	$O(\frac{1}{\epsilon})$, $O(\sum_{g\in\mathcal{G}} \lvert g\rvert \log \lvert g\rvert)$	$O(\frac{1}{\epsilon})$, $O(\sum_{g\in\mathcal{G}} \lvert g\rvert)$	$O(\frac{1}{\epsilon})$, $O(\sum_{g\in\mathcal{G}} \lvert g\rvert \log \lvert g\rvert)$

or ℓ_1/ℓ_∞ (Jenatton *et al.*, 2010; Liu and Ye, 2010b), and quadratic min-cost network flow for arbitrary overlapping groups with the ℓ_1/ℓ_∞ (Mairal *et al.*, 2010).

Table 2 summarizes the applicability, the convergence rate, and the per-iteration time complexity for the available first-order methods for different subclasses of group lasso penalties. More specifically, the methods in the first three rows adopt the proximal gradient framework. The first column of these rows gives the solver for the proximal operator. Each entry in Table 2 contains the convergence rate and the per-iteration time complexity. For the sake of simplicity, for all methods, we omit the time for computing the gradient of the loss function which is required for all of the methods (i.e., $\nabla g(\boldsymbol{\beta})$ with $O(J^2)$). The per-iteration time complexity in the table may come from the computation of proximal operator or subgradient of the penalty. "N.A." stands for "not applicable" or no guarantee in the convergence. As we can see from Table 2, although our method is not the most ideal one for some of the special cases, our method along with FOBOS (Duchi and Singer, 2009) are the only generic first-order methods that can be applied to all subclasses of the penalties.

As we can see from Table 2, for arbitrary overlaps with the ℓ_1/ℓ_∞, although the method proposed in Mairal *et al.* (2010) achieves $O(\frac{1}{\sqrt{\epsilon}})$ con-

vergence rate, the per-iteration complexity can be high due to solving a quadratic min-cost network flow problem. From the worst-case analysis, the per-iteration time complexity for solving the network flow problem in Mairal *et al.* (2010) is at least $O(|V||E|) = O((J+|\mathcal{G}|)(|\mathcal{G}|+J+\sum_{g\in\mathcal{G}}|g|))$, which is much higher than our method with $O(\sum_{g\in\mathcal{G}}|g|\log|g|)$. More importantly, for the case of arbitrary overlaps with the ℓ_1/ℓ_2, our method has a superior convergence rate to all the other methods.

In addition to these methods, an active-set algorithm was proposed that can be applied to the *square* of the ℓ_1/ℓ_2 mixed-norm with overlapping groups (Jenatton *et al.*, 2009). This method formulates each subproblem involving only the active variables 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. The latter approach involves an expensive matrix inversion at each iteration and lacks the global convergence rate. Another method (Liu and Ye, 2010a) was proposed for overlapping group lasso which approximately solves the proximal operator. However, the convergence of this type of approach cannot be guaranteed, since the error introduced in each proximal operator will be accumulated over iterations.

4.3. *Related work for fused lasso.* For the graph-guided-fused-lasso penalty, when the structure is a simple chain, the pathwise coordinate descent method (Friedman *et al.*, 2007) can be applied. For the general graph structure, a first-order method that approximately solves the proximal operator was proposed in Liu *et al.* (2010). However, the convergence cannot be guaranteed due to the errors introduced in computing the proximal operator over iterations.

Recently, two different path algorithms have been proposed (Tibshirani and Taylor, 2010; Zhou and Lange, 2011) that can be used to solve the graph-guided fused lasso as a special case. Unlike the traditional optimization methods that solve the problem for a fixed regularization parameter, they solve the entire path of solutions, and thus, has great practical advantages. In addition, for both methods, updating solutions from one hitting time to another is computationally very cheap. More specifically, a QR decomposition based updating scheme was proposed in Tibshirani and Taylor (2010) and the updating in Zhou and Lange (2011) can be done by an efficient sweep operation.

However, for high-dimensional data with $J \gg N$, the path algorithms can have the following problems:

1. For a general design matrix $\mathbf{X}$ other than the identity matrix, the

TABLE 3
Comparisons of different methods for optimizing graph-guided fused lasso

Method & Condition	Pre-processing Time	Per-iteration Time Complexity	No. of Iterations
H. Zhou & K. Lange (Zhou and Lange, 2011) ($\mathbf{X}$ full column rank, entire path)	$O(J^3)$	$O\left((\lvert E\rvert + J)^2\right)$	$O(\lvert E\rvert + J)$
R. Tibshirani & J. Taylor (Tibshirani and Taylor, 2010) ($\mathbf{X}$ full column rank, entire path)	$O\left(J^3 + N(\lvert E\rvert + J)\min((\lvert E\rvert + J), N)\right)$	$O\left(\min((\lvert E\rvert + J)^2, N^2)\right)$	$O(\lvert E\rvert + J)$ (lower bound)
R. Tibshirani & J. Taylor (Tibshirani and Taylor, 2010) ($\mathbf{X}$ not full column rank, entire path)	$O\left(J^3 + J^2N + (\lvert E\rvert + J)^2N\right)$	$O(N^2)$	$O(\lvert E\rvert + J)$ (lower bound)
SPG (single regularization parameter)	$O(NJ^2)$	$O(J^2 + \lvert E\rvert)$	$O(\frac{1}{\epsilon})$

method in Tibshirani and Taylor (2010) needs to first compute the pseudo-inverse of $\mathbf{X}$: $\mathbf{X}^+ = (\mathbf{X}^T\mathbf{X})^+\mathbf{X}^T$, which could be computationally expensive for large J.

2. The original version of the algorithms in Tibshirani and Taylor (2010); Zhou and Lange (2011) requires that $\mathbf{X}$ has a full column rank. When $J > N$, although one can add an extra $\epsilon\|\boldsymbol{\beta}\|_2^2$ term, this changes the original objective value especially when ϵ is large. For smaller ϵ, the matrix $(\mathbf{X}^*)^T\mathbf{X}^*$ with $\mathbf{X}^* = \begin{bmatrix} \mathbf{X} \\ \epsilon I \end{bmatrix}$ is highly ill-conditioned; and hence computing its inverse as the initilization step in Tibshirani and Taylor (2010) is very difficult. There is no known result on how to balance this trade-off.
3. In both Tibshirani and Taylor (2010) and Zhou and Lange (2011), the authors extend their algorithm to deal with the case when $\mathbf{X}$ does not have a full column rank. The extended version requires a Gramm-Schmidt process as the initialization which could take some extra time.

In Table 3, we present the comparisons for different methods. From our analysis, the method in Zhou and Lange (2011) is more efficient than the one in Tibshirani and Taylor (2010) since it avoids the heavy computation of the pseudo-inverse of $\mathbf{X}$. In practice, if $\mathbf{X}$ has a full column rank and one is interested in solutions on the entire path, the method in Zhou and Lange (2011) is very efficient and faster than our method. Instead, when $J \gg N$,

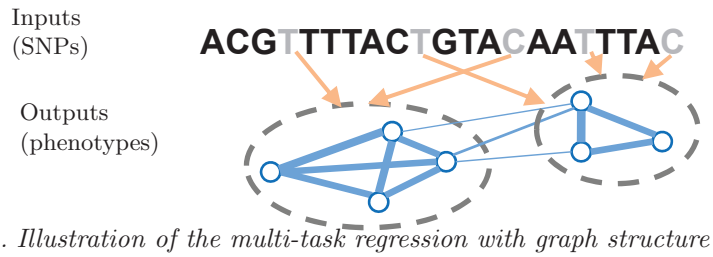


FIG 2. *Illustration of the multi-task regression with graph structure on outputs.*

the path following methods may require a time-consuming pre-processing procedure.

5. Extensions to Multi-task Regression with Structures on Outputs. The structured sparsity-inducing penalties as discussed in the previous section can be similarly used in the multi-task regression setting (Kim and Xing, 2010; Kim *et al.*, 2009) where the prior structural information is available for the outputs instead of inputs. 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 the phenotypes can be naturally represented as a graph, which can be used to guide the selection of SNPs as shown in Figure 2. Then, the graph-guided-fused-lasso penalty can be used to identify SNPs that are relevant jointly to multiple related phenotypes.

In a sparse multi-task regression with structure on the output side, we encounter the same difficulties of optimizing with non-smooth and non-separable penalties as in the previous section, and the SPG can be extended to this problem in a straightforward manner. Due to the importance of this class of problems and its applications, in this section, we briefly discuss how our method can be applied to the multi-task regression with structured-sparsity-inducing penalties.

5.1. *Multi-task Linear Regression Regularized by Structured Sparsity-inducing Penalties.* For the simplicity of illustration, we assume all different tasks share the same input matrix. 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 output: $\mathbf{y}_k = \mathbf{X}\boldsymbol{\beta}_k + \boldsymbol{\epsilon}_k, \quad \forall k = 1, \dots K$, where $\boldsymbol{\beta}_k = [\beta_{1k}, \dots, \beta_{Jk}]^T$ is the regression coefficient vector for the k-th output 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 outputs. Then, the multi-task (or multivariate-response) struc-

tured sparse regression problem can be naturally formulated as the following optimization problem:

$$\min_{\mathbf{B}\in\mathbb{R}^{J\times K}} f(\mathbf{B}) \equiv \frac{1}{2}\|\mathbf{Y}-\mathbf{XB}\|_F^2 + \Omega(\mathbf{B}) + \lambda\|\mathbf{B}\|_1, \tag{5.1}$$

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 a structure over the outputs.

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

$$\Omega(\mathbf{B}) \equiv \gamma \sum_{j=1}^{J} \sum_{g\in\mathcal{G}} w_g \|\boldsymbol{\beta}_{jg}\|_2, \tag{5.2}$$

   where $\mathcal{G} = \{g_1, \ldots, g_{|\mathcal{G}|}\}$ is a subset of the power set of $\{1, \ldots, K\}$ and $\boldsymbol{\beta}_{jg}$ is the vector of regression coefficients correspond to outputs in group g: $\{\beta_{jk}, k \in g, g \in \mathcal{G}\}$. Both the ℓ_1/ℓ_2 mixed-norm penalty for multi-task regression in Obozinski *et al.* (2009) and tree-structured overlapping-group-lasso penalty in Kim and Xing (2010) are special cases of (5.2).
2. **Graph-guided-fused-lasso Penalty in Multi-task Regression** Assuming that a graph structure over the K outputs is given as G with a set of nodes $V = \{1, \ldots, K\}$ each corresponding to an output variable and a set of edges E, the graph-guided-fused-lasso penalty for a structured multi-task regression is given as:

$$\Omega(\mathbf{B}) = \gamma \sum_{e=(m,l)\in E} \tau(r_{ml}) \sum_{j=1}^{J} |\boldsymbol{\beta}_{jm} - \mathrm{sign}(r_{ml})\boldsymbol{\beta}_{jl}|. \tag{5.3}$$

5.2. *Smoothing Proximal Gradient Descent.* Using the similar techniques in Section 3.1, $\Omega(\mathbf{B})$ can be reformulated as:

$$\Omega(\mathbf{B}) = \max_{\mathbf{A}\in\mathcal{Q}} \langle C\mathbf{B}^T, \mathbf{A}\rangle, \tag{5.4}$$

where $\langle \mathbf{U}, \mathbf{V}\rangle \equiv \mathrm{Tr}(\mathbf{U}^T\mathbf{V})$ denotes a matrix inner product. C is constructed in the similar way as in (3.2) or (3.3) just by replacing the index of the input variables with the output variables, and $\mathbf{A}$ is the matrix of auxiliary variables.

TABLE 4
Comparison of Per-iteration Time Complexity for Multi-task Regression

	Overlapping Group Lasso	Graph-guided Fused Lasso
SPG	$O(JK\min(J,N)+J\sum_{g\in\mathcal{G}}\|g\|)$	$O(JK\min(J,N)+J\|E\|)$
IPM	$O\left(J^2(K+\|\mathcal{G}\|)^2(KN+J(\sum_{g\in\mathcal{G}}\|g\|))\right)$	$O\left(J^3(K+\|E\|)^3\right)$

Then we introduce the smooth approximation of (5.4):

$$f_\mu(\mathbf{B})=\max_{\mathbf{A}\in\mathcal{Q}}\left(\langle C\mathbf{B}^T,\mathbf{A}\rangle-\mu d(\mathbf{A})\right), \tag{5.5}$$

where $d(\mathbf{A})\equiv\frac{1}{2}\|\mathbf{A}\|_F^2$. Following a proof strategy similar to that in Theorem 1, we can show that $f_\mu(\mathbf{B})$ is convex and smooth with gradient $\nabla f_\mu(\mathbf{B})=(\mathbf{A}^*)^TC$, where $\mathbf{A}^*$ is the optimal solution to (5.5). The closed-form solution of $\mathbf{A}^*$ and the Lipschitz constant for $\nabla f_\mu(\mathbf{B})$ can be derived in the same way.

By substituting $\Omega(\mathbf{B})$ in (5.1) with $f_\mu(\mathbf{B})$, we can adopt Algorithm 1 to solve (5.1) with convergence rate of $O(\frac{1}{\epsilon})$. The per-iteration time complexity of SPG as compared to IPM for SOCP or QP formulation is presented int Table 4. As we can see, the per-iteration complexity for SPG is linear in $\max(|K|,\sum_{g\in\mathcal{G}}|g|)$ or $\max(|K|,|E|)$ while traditional approaches based on IPM scape at least cubically to the size of outputs K.

6. Experiment. In this section, we evaluate the scalability and efficiency of the smoothing proximal gradient method (SPG) on a number of structured sparse regression problems via simulation, and apply SPG to an overlapping group lasso problem on a real genetic datable.

On an overlapping group lasso problem, we compare the SPG with FOBOS (Duchi and Singer, 2009) and IPM for SOCP.[3] On a multi-task graph-guided fused lasso problem, we compare the running time of SPG with that of the FOBOS (Duchi and Singer, 2009) and IPM for QP.[4] Note that for FOBOS, since the proximal operator associated with $\Omega(\boldsymbol{\beta})$ cannot be solved exactly, we set the "loss function" to $l(\boldsymbol{\beta})=g(\boldsymbol{\beta})+\Omega(\boldsymbol{\beta})$ and the penalty to $\lambda\|\boldsymbol{\beta}\|_1$. According to Duchi and Singer (2009), for the non-smooth loss $l(\boldsymbol{\beta})$, FOBOS achieves $O\left(\frac{1}{\epsilon^2}\right)$ convergence rate, which is slower than our method.

All experiments are performed on a standard PC with 4GB RAM and the software is written in MATLAB. The main difficulty in comparisons is

[3]We use the state-of-the-art MATLAB package SDPT3 (Tütüncü *et al.*, 2003) for SOCP.

[4]We use the commercial package MOSEK (`http://www.mosek.com/`) for QP. Graph-guided fused lasso can also be solved by SOCP but it is less efficient than QP.

a fair stopping criterion. Unlike IPM, SPG and FOBOS do not generate a dual solution, and therefore, it is not possible to compute a primal-dual gap, which is the traditional stopping criterion for IPM. Here, we adopt a widely used approach for comparing different methods in optimization literature. Since it is well known that IPM usually gives more accurate (i.e., lower) objective, we set the objective obtained from IPM as the optimal objective value and stop the first-order methods when the objective is below 1.001 times the optimal objective. For large datasets for which IPM cannot be applied, we stop the first-order methods when the relative change in the objective is below 10^{-6}. In addition, maximum iterations is set to 20,000.

Since our main focus is on the optimization algorithm, for the purpose of simplicity, we assume that each group in the overlapping group lasso problem receives the same amount of regularization and hence set the weights w_g for all group to be 1. In principle, more sophisticated prior knowledge of the importance for each group can be naturally incorporated into w_g. In addition, we notice that each variable j with the regularization $\lambda|\beta_j|$ in $\lambda\|\boldsymbol{\beta}\|_1$ can be viewed as a singleton group. To ease the tuning of parameters, we again assume that each group (including the singleton group) receives the same amount of regularization and hence constrain the regularization parameters $\lambda = \gamma$.

The smoothing parameter μ is set to $\frac{\epsilon}{2D}$ according to Theorem 2, where D is determined by the problem size. It is natural that for large-scale problems with large D, a larger ϵ can be adopted without affecting the recovery quality significantly. Therefore, instead of setting ϵ, we directly set $\mu = 10^{-4}$, which provided us with reasonably good approximation accuracies for different scales of problems based on our experience for a range of μ in simulations. As for FOBOS, we set the stepsize rate to $\frac{c}{\sqrt{t}}$ as suggested in Duchi and Singer (2009), where c is carefully tuned to be $\frac{0.1}{\sqrt{NJ}}$ for univariate regression and $\frac{0.1}{\sqrt{NJK}}$ for multi-task regression.

6.1. *Simulation Study I: Overlapping Group Lasso.* We simulate data for a univariate linear regression model with the overlapping group structure on the inputs as described below. Assuming that the inputs are ordered, we define a sequence of groups of 100 adjacent inputs with an overlap of 10 variables between two successive groups so that

$$\mathcal{G} = \{\{1, \ldots, 100\}, \{91, \ldots, 190\}, \ldots, \{J - 99, \ldots, J\}\},$$

with $J = 90|\mathcal{G}| + 10$. We set $\beta_j = (-1)^j \exp(-(j-1)/100)$ for $1 \le j \le J$. We sample each element of $\mathbf{X}$ from *i.i.d.* Gaussian distribution, and generate the output data from $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim N(0, I_{N\times N})$.

TABLE 5
Comparisons of different optimization methods on the overlapping group lasso

$\|\mathcal{G}\| = 10$		N=1,000		N=5,000		N=10,000	
($J = 910$)		CPU (s)	Obj.	CPU (s)	Obj.	CPU (s)	Obj.
$\gamma = 2$	SOCP	103.71	266.683	493.08	917.132	3777.46	1765.518
	FOBOS	27.12	266.948	1.71	918.019	1.48	1765.613
	SPG	0.87	266.947	0.71	917.463	1.28	1765.692
$\gamma = 0.5$	SOCP	106.02	83.304	510.56	745.102	3585.77	1596.418
	FOBOS	32.44	82.992	4.98	745.788	4.65	1597.531
	SPG	0.42	83.386	0.41	745.104	0.69	1596.452
$\|\mathcal{G}\| = 50$		N=1,000		N=5,000		N=10,000	
($J = 4510$)		CPU (s)	Obj.	CPU (s)	Obj.	CPU (s)	Obj.
$\gamma = 10$	SOCP	4144.20	1089.014	-	-	-	-
	FOBOS	476.91	1191.047	394.75	1533.314	79.82	2263.494
	SPG	56.35	1089.052	77.61	1533.318	78.90	2263.601
$\gamma = 2.5$	SOCP	3746.43	277.911	-	-	-	-
	FOBOS	478.62	286.327	867.94	559.251	183.72	1266.728
	SPG	33.09	277.942	30.13	504.337	26.74	1266.723
$\|\mathcal{G}\| = 100$		N=1,000		N=5,000		N=10,000	
($J = 9010$)		CPU (s)	Obj.	CPU (s)	Obj.	CPU (s)	Obj.
$\gamma = 20$	FOBOS	1336.72	2090.808	2261.36	3132.132	1091.20	3278.204
	SPG	234.71	2090.792	225.28	2692.981	368.52	3278.219
$\gamma = 5$	FOBOS	1689.69	564.209	2287.11	1302.552	3342.61	1185.661
	SPG	169.61	541.611	192.92	736.559	176.72	1114.933

To demonstrate the efficiency and scalability of SPG, we vary J, N and γ and report the total CPU time in seconds and the objective value in Table 5. The regularization parameter γ is set to either $|\mathcal{G}|/5$ or $|\mathcal{G}|/20$. As we can see from Table 5, firstly, both SPG and FOBOS are more efficient and scalable by orders of magnitude than IPM for SOCP. For larger J and N, we are unable to collect the results for SOCP. Secondly, SPG is more efficient than FOBOS

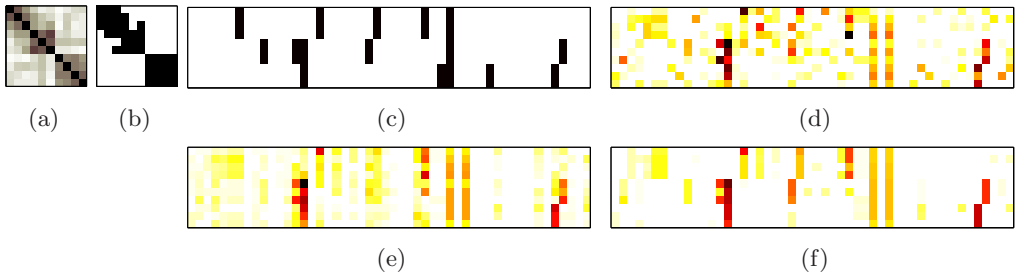


FIG 3. *Regression coefficients estimated by different methods based on a single simulated datable.* $b = 0.8$ *and threshold* $\rho = 0.3$ *for the output correlation graph are used. Red pixels indicate large values. (a) The correlation coefficient matrix of phenotypes, (b) the edges of the phenotype 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 regression, (f) Graph-guided fused lasso. Rows correspond to outputs and columns to inputs.*

for almost all different scales of the problems.[5] Thirdly, for SPG, a smaller γ leads to faster convergence. This result is consistent with Theorem 2, which shows that the number of iterations is linear in γ through the term $\|C\|$. Moreover, we notice that a larger N does not increase the computational time for SPG. This is also consistent with the time complexity analysis, which shows that for linear regression, the per-iteration time complexity is independent of N.

However, we find that the solutions from IPM are more accurate and in fact, it is hard for first-order approaches to achieve the same precision as IPM. Assuming that we require $\epsilon = 10^{-6}$ for the accuracy of the solution, it takes IPM about $O(\log(\frac{1}{\epsilon})) \approx 14$ iterations to converge while $O(\frac{1}{\epsilon}) = 10^6$ iterations for SPG. This is the drawback for any first-order method. However, in many real applications, we do not require the objective to be extremely accurate (e.g., $\epsilon = 10^{-3}$ is sufficiently accurate in general) and first-order methods are more suitable. More importantly, first-order methods can be applied to large-scale high-dimensional problems while IPM can only be applied to small or moderate scale problems due to the expensive computation necessary for solving the Newton linear system.

[5]In some entries in Table 5, the Obj. from FOBOS is much larger than other methods. This is because that FOBOS has reached the maximum number of iterations before convergence. Instead, for our simulations, SPG generally converges in hundreds or at most, a few thousands, iterations and never pre-terminates.

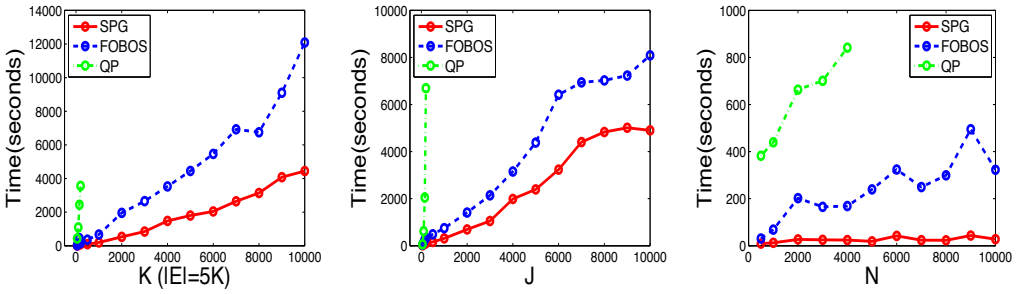


FIG 4. *Comparisons of SPG, FOBOS and QP. (a) Vary K from* 50 *to* $10,000$*, fixing* $N = 500, J = 100$*; (b) Vary J from* 50 *to* $10,000$*, fixing* $N = 1000, K = 50$*; and (c) Vary N from* 500 *to* 10000*, fixing* $J = 100, K = 50$.

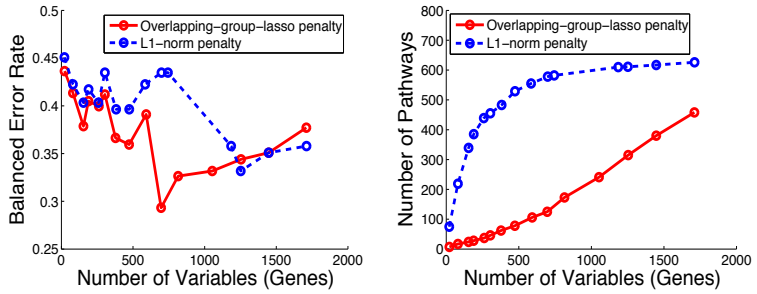
6.2. *Simulation Study II: Multi-task Graph-guided Fused Lasso.* We simulate data using the following scenario analogous to the problem of genetic association mapping, where we are interested in identifying a small number of genetic variations (inputs) that influence the phenotypes (outputs). We use $K = 10$, $J = 30$ 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 $\boldsymbol{\beta}_k$'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 3(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 3(a) at $\rho = 0.3$ to obtain the graph in Figure 3(b), and use this graph as prior structural information for graph-guided fused lasso. As an illustrative example, the estimated regression coefficients from different regression models for recovering the association patterns are shown in Figures 3(d)–(f). While the results of lasso and ℓ_1/ℓ_2-regularized multi-task regression with $\Omega(\mathbf{B}) = \sum_{j=1}^{J} \|\boldsymbol{\beta}_{j,:}\|_2$

(Obozinski *et al.*, 2009) in Figures 3 (d) and (e) contain many false positives, the results from graph-guided fused lasso in Figure 3(f) show fewer false positives and reveal clear block structures. Thus, the graph-guided fused lasso proves to be a superior regression model for recovering the true regression pattern that involves structured sparsity in the input/output relationships.

To compare SPG with FOBOS and IPM for QP in solving such a structured sparse regression problem, we vary K, J, N, and present the computation time in seconds in Figures 4(a)-(c), respectively. We select the regularization parameter γ using a separate validation datable, and report the CPU time for graph-guided fused lasso with the selected γ. The input/output data and true regression coefficient matrix $\mathbf{B}$ are generated in the way similar as 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. We set the ρ for each datable so that the number of edges is 5 times the number of the nodes (i.e. $|E| = 5K$). Figure 4 shows that SPG is substantially more efficient and can scale up to very high-dimensional and large-scale datasets. Moreover, we notice that the increase of N almost does not affect the computation time of SPG, which is consistent with the complexity analysis in Section 3.5.

6.3. *Real Data Analysis: Pathway Analysis of Breast Cancer Data.* In this section, we apply the SPG to an overlapping group lasso problem with a logistic loss on a real-world datable collected from breast cancer tumors (van de Vijver et al., 2002; Jacob *et al.*, 2009).The main goal is to demonstrate the importance of employing structured sparsity-inducing penalties for performance enhancement in real life high-dimensional regression problems, thereby further exhibit and justify the needs of efficient solvers such as SPG for such problems.

The data are given as gene expression measurements for 8,141 genes in 295 breast-cancer tumors (78 metastatic and 217 non-metastatic). A lot of research efforts in biology have been devoted to identifying biological pathways that consist of a group of genes participating in a particular biological process to perform a certain functionality in the cell. 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



(a) (b)

FIG 5. *Results from the analysis of breast cancer datable. (a) Balanced error rate for varying the number of selected genes, and (b) the number of pathways for varying the number of selected genes.*

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. Instead of analyzing all 8,141 genes, we focus on these 3,510 genes which belong to certain pathways. 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 SPG.

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.[6] 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 5, we compare the results from fitting the logistic regression with the overlapping-group-lasso penalty with a baseline model with only the ℓ_1-norm penalty. Figure 5(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

[6]See http://www.modelselect.inf.ethz.ch/evaluation.php for more details.

penalty is lower than the one with the 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 5(b), for the different numbers of selected genes, we show the number of pathways to which the selected genes belong. From Figure 5(b), we see that when the group structure information is incorporated, fewer pathways are selected. This indicates that regression 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 for computing the whole regularization path with 20 different values for the regularization parameters is 331 seconds for the overlapping group lasso.

We perform 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 5(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 overlapping group lasso 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 5. 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 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 3.3E-13) and the process of *oxidation reduction* (73 genes with p-value 1.7E-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. Conclusions and Future Work. In this paper, we investigated an optimization problem for estimating the structured-sparsity pattern in regression coefficients under a general class of structured sparsity-inducing penalties. Many of the structured sparsity-inducing penalties including the overlapping-group-lasso penalties and graph-guided-fused-lasso 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 a common form, and proposed a general optimization approach called smoothing proximal gradient method for efficiently solving the optimization problem of this common form. Our results show that the proposed method enjoys both desirable theoretical guarantee and practical scalability under various difficult settings involving complex structure constraints, multi-task, and high-dimensionality.

There are several future directions for this work. Firstly, it is known that reducing μ over iterations leads to better empirical results. However, in such a scenario, the convergence rate is harder to analyze. Moreover, 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.

Another interesting direction is to incorporate other accelerating techniques into our method to further boost the performance. For example, the technique introduced in Zhou *et al.* (2011) can efficiently accelerate the algorithms which essentially solve a fixed point problem as $\boldsymbol{\beta} = F(\boldsymbol{\beta})$. It uses an approximation of the Jacobian of $F(\boldsymbol{\beta})$. It is very interesting to incorporate this technique into our framework. However, since there is an ℓ_1-norm

penalty in our model and the operator F is hence non-differentiable, it is difficult to compute the approximation of the Jacobian of F. One potential strategy is to use the idea from semi-smooth Newton method (Qi and Sun, 1993; Sun *et al.*, 2002) to solve the non-smooth operator F.

8. Appendix.

8.1. *Proof of Theorem 1.* We first introduce the concept of *Fenchel Conjugate*.

DEFINITION 1. *The Fenchel conjugate of a function $\varphi(\boldsymbol{\alpha})$ is the function $\varphi^*(\boldsymbol{\beta})$ defined as:*

$$\varphi^*(\boldsymbol{\beta}) = \sup_{\boldsymbol{\alpha} \in dom(\varphi)} \left(\boldsymbol{\alpha}^T \boldsymbol{\beta} - \varphi(\boldsymbol{\alpha}) \right).$$

Recall that $d(\boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{\alpha}\|^2$ with the $\text{dom}(\boldsymbol{\alpha}) = \mathcal{Q}$. According to Definition 1, the conjugate of $d(\cdot)$ at $\frac{C\boldsymbol{\beta}}{\mu}$ is: $d^*\left(\frac{C\boldsymbol{\beta}}{\mu}\right) = \sup_{\boldsymbol{\alpha} \in \mathcal{Q}} \left(\boldsymbol{\alpha}^T \frac{C\boldsymbol{\beta}}{\mu} - d(\boldsymbol{\alpha}) \right)$, and hence

$$f_\mu(\boldsymbol{\beta}) \equiv \arg\max_{\boldsymbol{\alpha} \in \mathcal{Q}} \left(\boldsymbol{\alpha}^T C\boldsymbol{\beta} - \mu d(\boldsymbol{\alpha}) \right) = \mu d^* \left(\frac{C\boldsymbol{\beta}}{\mu} \right).$$

According to Theorem 26.3 in Rockafellar (1996) "a closed proper convex function is essentially strictly convex if and only if its conjugate is essentially smooth", since $d(\boldsymbol{\alpha})$ is a closely proper strictly convex function, its conjugate is smooth. Therefore, $f_\mu(\boldsymbol{\beta})$ is a smooth function.

Now we apply Danskin's Theorem (Prop B.25 in Bertsekas (1999)) to derive $\nabla f_\mu(\boldsymbol{\beta})$. Let $\phi(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \boldsymbol{\alpha}^T C\boldsymbol{\beta} - \mu d(\boldsymbol{\alpha})$. Since $d(\cdot)$ is a strongly convex function, $\arg\max_{\boldsymbol{\alpha} \in \mathcal{Q}} \phi(\boldsymbol{\alpha}, \boldsymbol{\beta})$ has a unique optimal solution and we denote it as $\boldsymbol{\alpha}^*$. According to Danskin's Theorem:

$$\nabla f_\mu(\boldsymbol{\beta}) = \nabla_{\boldsymbol{\beta}} \phi(\boldsymbol{\alpha}^*, \boldsymbol{\beta}) = C^T \boldsymbol{\alpha}^*. \tag{8.1}$$

As for the proof of Lipschitz constant of $f_\mu(\boldsymbol{\beta})$, readers may refer to Nesterov (2005).

8.2. *Proof of Proposition 1.*

$$\begin{aligned} \boldsymbol{\alpha}^* &= \arg\max_{\boldsymbol{\alpha} \in \mathcal{Q}} \left(\boldsymbol{\alpha}^T C\boldsymbol{\beta} - \frac{\mu}{2}\|\boldsymbol{\alpha}\|_2^2 \right) \\ &= \arg\max_{\boldsymbol{\alpha} \in \mathcal{Q}} \sum_{g \in \mathcal{G}} \left(\gamma w_g \boldsymbol{\alpha}_g^T \boldsymbol{\beta}_g - \frac{\mu}{2}\|\boldsymbol{\alpha}_g\|_2^2 \right) \\ &= \arg\min_{\boldsymbol{\alpha} \in \mathcal{Q}} \sum_{g \in \mathcal{G}} \|\boldsymbol{\alpha}_g - \frac{\gamma w_g \boldsymbol{\beta}_g}{\mu}\|_2^2 \end{aligned} \tag{8.2}$$

Therefore, (8.2) can be decomposed into $|\mathcal{G}|$ independent problems: each one is the Euclidean projection onto the ℓ_2-ball:

$$\boldsymbol{\alpha}_g^* = \underset{\boldsymbol{\alpha}_g:\|\boldsymbol{\alpha}_g\|_2\leq 1}{\arg\min} \|\boldsymbol{\alpha}_g - \frac{\gamma w_g \boldsymbol{\beta}_g}{\mu}\|_2^2,$$

and $\boldsymbol{\alpha}^* = \left[(\boldsymbol{\alpha}_{g_1}^*)^T, \ldots, (\boldsymbol{\alpha}_{g_{|\mathcal{G}|}}^*)^T\right]^T$. According to the property of ℓ_2-ball, it can be easily shown that:

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

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

As for $\|C\|$,

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

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

$$\|C\mathbf{v}\|_2 = \gamma \max_{j\in\{1,\ldots,J\}} \sqrt{\sum\nolimits_{g\in\mathcal{G}\ \text{s.t.}\ j\in g}(w_g)^2}.$$

8.3. *Proof of Proposition 2.* Similar to the proof technique of Proposition 1, we reformulate the problem of solving $\boldsymbol{\alpha}^*$ as a Euclidean projection:

$$\boldsymbol{\alpha}^* = \underset{\boldsymbol{\alpha}\in\mathcal{Q}}{\arg\max}\left(\boldsymbol{\alpha}^T C\boldsymbol{\beta} - \frac{\mu}{2}\|\boldsymbol{\alpha}\|_2^2\right) = \underset{\boldsymbol{\alpha}:\|\boldsymbol{\alpha}\|_\infty\leq 1}{\arg\min} \|\boldsymbol{\alpha} - \frac{C\boldsymbol{\beta}}{\mu}\|_2^2,$$

and the optimal solution $\boldsymbol{\alpha}^*$ can be obtained by projecting $\frac{C\boldsymbol{\beta}}{\mu}$ onto the ℓ_∞-ball.

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

$$\|C\mathbf{v}\|_2^2 = \gamma^2 \sum_{e=(m,l)\in E} (\tau(r_{ml}))^2 (v_m - \text{sign}(r_{ml})v_l)^2 \tag{8.3}$$

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 (8.3) can be further bounded by:

$$\begin{aligned} \|C\mathbf{v}\|_2^2 &\le \gamma^2 \textstyle\sum_{e=(m,l)\in E} 2(\tau(r_{ml}))^2(v_m^2 + v_l^2) \\ &= \gamma^2 \textstyle\sum_{j\in V}(\sum_{e \text{ incident on } k} 2(\tau(r_e))^2)v_j^2 \\ &= \gamma^2 \textstyle\sum_{j\in V} 2d_j v_j^2 \\ &\le 2\gamma^2 \max_{j\in V} d_j, \end{aligned} \tag{8.4}$$

where

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

Therefore, we have

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

Note that this upper bound is tight because the first inequality in (8.4) is tight.

8.4. *Proof of Theorem 2.* Based on the result from Beck and Teboulle (2009), we have the following lemma:

LEMMA 1. *For the function $\widetilde{f}(\boldsymbol{\beta}) = h(\boldsymbol{\beta}) + \lambda\|\boldsymbol{\beta}\|_1$, 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 $\widetilde{f}(\boldsymbol{\beta})$ and let $\boldsymbol{\beta}^t$ be the approximate solution at the t-th iteration. For any $\boldsymbol{\beta}$, we have the following bound:*

$$\widetilde{f}(\boldsymbol{\beta}^t) - \widetilde{f}(\boldsymbol{\beta}) \le \frac{2L\|\boldsymbol{\beta} - \boldsymbol{\beta}^0\|_2^2}{t^2}. \tag{8.5}$$

In order to use the bound in (8.5), we use the similar proof scheme as in Lan *et al.* (2011) and decompose $f(\boldsymbol{\beta}^t) - f(\boldsymbol{\beta}^*)$ into three terms:

$$f(\boldsymbol{\beta}^t) - f(\boldsymbol{\beta}^*) = \left(f(\boldsymbol{\beta}^t) - \widetilde{f}(\boldsymbol{\beta}^t)\right) + \left(\widetilde{f}(\boldsymbol{\beta}^t) - \widetilde{f}(\boldsymbol{\beta}^*)\right) + \left(\widetilde{f}(\boldsymbol{\beta}^*) - f(\boldsymbol{\beta}^*)\right). \tag{8.6}$$

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

$$\widetilde{f}(\boldsymbol{\beta}) \le f(\boldsymbol{\beta}) \le \widetilde{f}(\boldsymbol{\beta}) + \mu D,$$

where $D \equiv \max_{\boldsymbol{\alpha}\in\mathcal{Q}} d(\boldsymbol{\alpha})$. Therefore, the first term in (8.6), $f(\boldsymbol{\beta}^t) - \widetilde{f}(\boldsymbol{\beta}^t)$, is upper-bounded by μD, and the last term in (8.6) is less than or equal to

0 (i.e., $\widetilde{f}(\boldsymbol{\beta}^*) - f(\boldsymbol{\beta}^*) \leq 0$). Combining (8.5) with these two simple bounds, we have:

(8.7)
$$f(\boldsymbol{\beta}^t) - f(\boldsymbol{\beta}^*) \leq \mu D + \frac{2L\|\boldsymbol{\beta}^* - \boldsymbol{\beta}^0\|_2^2}{t^2} \leq \mu D + \frac{2\|\boldsymbol{\beta}^* - \boldsymbol{\beta}^0\|_2^2}{t^2}\left(\lambda_{\max}(\mathbf{X}^T\mathbf{X}) + \frac{\|C\|^2}{\mu}\right).$$

By setting $\mu = \frac{\epsilon}{2D}$ and plugging this into the right-hand side of (8.7), we obtain

$$f(\boldsymbol{\beta}^t) - f(\boldsymbol{\beta}^*) \leq \frac{\epsilon}{2} + \frac{2\|\boldsymbol{\beta}^*\|_2^2}{t^2}\left(\lambda_{\max}\left(\mathbf{X}^T\mathbf{X}\right) + \frac{2D\|C\|^2}{\epsilon}\right). \tag{8.8}$$

If we require the right-hand side of (8.8) to be equal to ϵ and solve it for t, we obtain the bound of t in (3.16).

Acknowledgements. We would like to thank Yanjun Qi for the help of preparation and verification of breast cancer datable and Javier Peña for the discussion of the related first-order methods. We would also like to thank anonymous reviewers and the associate editor for their constructive comments on improving the quality of the paper.

References.

Abate, N., Chandalia, M., Satija, P., Adams-Huet, B., and et. al. (2005). Enpp1/pc-1 k121q polymorphism and genetic susceptibility to type 2 diabetes. *Diabetes*, **54(4)**, 1027–1213.

Beck, A. and Teboulle, M. (2009). A fast iterative shrinkage thresholding algorithm for linear inverse problems. *SIAM Journal of Image Science*, **2(1)**, 183–202.

Bertsekas, D. (1999). *Nonlinear Programming*. Athena Scientific.

Duchi, J. and Singer, Y. (2009). Efficient online and batch learning using forward backward splitting. *Journal of Machine Learning Research*, **10**, 2899–2934.

Friedman, J., Hastie, T., Höfling, H., and Tibshirani, R. (2007). Pathwise coordinate optimization. *Annals of Applied Statistics*, **1**, 302–332.

Friedman, J., Hastie, T., and Tibshirani, R. (2010). A note on the group lasso and a sparse group lasso.

Huang, D. W., Sherman, B. T., and Lempicki, R. A. (2009). Systematic and integrative analysis of large gene lists using david bioinformatics resources. *Nature Protoc*, **4(1)**, 44–57.

Jacob, L., Obozinski, G., and Vert, J.-P. (2009). Group lasso with overlap and graph lasso. In *Proceedings of the International Conference on Machine Learning*.

Jenatton, R., Audibert, J.-Y., and Bach, F. (2009). Structured variable selection with sparsity-inducing norms. Technical report, INRIA.

Jenatton, R., Mairal, J., Obozinski, G., and Bach, F. (2010). Proximal methods for sparse hierarchical dictionary learning. In *Proceedings of the International Conference on Machine Learning*.

Kim, S. and Xing, E. P. (2009). Statistical estimation of correlated genome associations to a quantitative trait network. *PLoS Genetics*, **5(8)**.

Kim, S. and Xing, E. P. (2010). Tree-guided group lasso for multi-task regression with structured sparsity. In *Proceedings of the International Conference on Machine Learning*.

Kim, S., Sohn, K.-A., and Xing, E. P. (2009). A multivariate regression approach to association analysis of a quantitative trait network. *Bioinformatics*, **25(12)**, 204–212.

Lan, G., Lu, Z., and Monteiro, R. (2011). Primal-dual first-order methods with $O(1/\epsilon)$ iteration complexity for cone programming. *Mathematical Programming*, **126**, 1–29.

Lange, K. (2004). *Optimization*. Springer.

Liu, J. and Ye, J. (2010a). Fast overlapping group lasso. ArXiv:1009.0306v1 [cs.LG].

Liu, J. and Ye, J. (2010b). Moreau-yosida regularization for grouped tree structure learning. In *Advances in Neural Information Processing Systems (NIPS)*.

Liu, J., Ji, S., and Ye, J. (2009). Multi-task feature learning via efficient $\ell_{2,1}$-norm minimization. In *Proceedings of the Uncertainty in AI*.

Liu, J., Yuan, L., and Ye, J. (2010). An efficient algorithm for a class of fused lasso problems. In *the ACM SIG Konwledge Discovery and Data Mining*.

Ma, S. and Kosorok, M. R. (2010). Detection of gene pathways with predictive power for breast cancer prognosis. *BMC Bioinformatics*, **11(1)**.

Mairal, J., Jenatton, R., Obozinski, G., and Bach, F. (2010). Network flow algorithms for structured sparsity. In *Advances in Neural Information Processing Systems (NIPS)*.

Nesterov, Y. (2003). Excessive gap technique in non-smooth convex minimization. Technical report, Universit catholique de Louvain, Center for Operations Research and Econometrics (CORE).

Nesterov, Y. (2005). Smooth minimization of non-smooth functions. *Mathematical Programming*, **103**(1), 127–152.

Nesterov, Y. (2007). Gradient methods for minimizing composite objective function. ECORE Discussion Paper 2007.

Obozinski, G., Taskar, B., and Jordan, M. I. (2009). High-dimensional union support recovery in multivariate regression. In *Advances in Neural Information Processing Systems (NIPS)*.

Qi, L. and Sun, J. (1993). A nonsmooth version of newton's method. *Mathematical Programming*, **58**, 353–367.

Rockafellar, R. (1996). *Convex Analysis*. Princeton Univ. Press.

Subramanian, A., Tamayo, P., Mootha, V., and et. al (2005). 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.

Sun, D., Womersley, R., and Qi, H. (2002). A feasible semismooth asymptotically newton method for mixed complementarity problems. *Mathematical Programming, Ser A*, **94**, 167–187.

The International HapMap Consortium (2005). A haplotype map of the human genome. *Nature*, **437**, 1399–1320.

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

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

Tibshirani, R. and Taylor, J. (2010). The solution path of the generalized lasso. *Annals of Statistics*, **39 (3)**, 1335–1371.

Tseng, P. (2008). On accelerated proximal gradient methods for convex-concave optimization. *SIAM Journal on Optimization (Submitted)*.

Tütüncü, R. H., Toh, K. C., and Todd, M. J. (2003). Solving semidefinite-quadratic-linear programs using sdpt3. *Mathematical Programming*, **95**, 189–217.

van de Vijver et al., M. J. (2002). A gene-expression signature as a predictor of survival in breast cancer. *New England Journal of Medicine*, **347**, 1999–2009.

Wu, T. and Lange, K. (2008). Coordinate descent algorithms for lasso penalized regression. *Annals of Applied Statistics*, **2**, 224–244.

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

Zhang, Z., Lange, K., Ophoff, R., and Sabatti, C. (2010). Reconstruction and dna copy number by penalized estimation and imputation. *Annals of Applied Statistics*, **4**, 1749–1773.

Zhao, P., Rocha, G., and Yu, B. (2009a). The composite absolute penalties family for grouped and hierarchical variable selection. *Annals of Statistics*, **37(6A)**, 3468–3497.

Zhao, P., Rocha, G., and Yu, B. (2009b). Grouped and hierarchical model selection through composite absolute penalties. *Annals of Statistics*, **37(6A)**, 3468–3497.

Zhou, H. and Lange, K. (2011). A path algorithm for constrained estimation. arXiv:1103.3738v1 [stat.CO].

Zhou, H., Alexander, D., and Lange, K. (2011). A quasi-newton acceleration for high-dimensional optimization algorithms. *Statistics and Computing*, **21**, 261–273.

Xi Chen
Seyoung Kim
Jaime G. Carbonell
Eric P. Xing
School of Computer Science
Carnegie Mellon University
E-mail: xichen@cs.cmu.edu
sssykim@cs.cmu.edu
jgc@cs.cmu.edu
epxing@cs.cmu.edu

Qihang Lin
Tepper School of Business
Carnegie Mellon University
E-mail: qihangl@andrew.cmu.edu