Efficient Methods for Overlapping Group Lasso

Lei Yuan
Arizona State University
Tempe, AZ, 85287
Lei.Yuan@asu.edu

Jun Liu
Arizona State University
Tempe, AZ, 85287
j.liu@asu.edu

Jieping Ye
Arizona State University
Tempe, AZ, 85287
jieping.ye@asu.edu

Abstract

The group Lasso is an extension of the Lasso for feature selection on (predefined) non-overlapping groups of features. The non-overlapping group structure limits its applicability in practice. There have been several recent attempts to study a more general formulation, where groups of features are given, potentially with overlaps between the groups. The resulting optimization is, however, much more challenging to solve due to the group overlaps. In this paper, we consider the efficient optimization of the overlapping group Lasso penalized problem. We reveal several key properties of the proximal operator associated with the overlapping group Lasso, and compute the proximal operator by solving the smooth and convex dual problem, which allows the use of the gradient descent type of algorithms for the optimization. We have performed empirical evaluations using both synthetic and the breast cancer gene expression data set, which consists of 8,141 genes organized into (overlapping) gene sets. Experimental results show that the proposed algorithm is more efficient than existing state-of-the-art algorithms.

1 Introduction

Problems with high dimensionality have become common over the recent years. The high dimensionality poses significant challenges in building interpretable models with high prediction accuracy. Regularization has been commonly employed to obtain more stable and interpretable models. A well-known example is the penalization of the ℓ_1 norm of the estimator, known as Lasso [25]. The ℓ_1 norm regularization has achieved great success in many applications. However, in some applications [28], we are interested in finding important explanatory factors in predicting the response variable, where each explanatory factor is represented by a group of input features. In such cases, the selection of important features corresponds to the selection of groups of features. As an extension of Lasso, group Lasso [28] based on the combination of the ℓ_1 norm and the ℓ_2 norm has been proposed for group feature selection, and quite a few efficient algorithms [16, 17, 19] have been proposed for efficient optimization. However, the non-overlapping group structure in group Lasso limits its applicability in practice. For example, in microarray gene expression data analysis, genes may form overlapping groups as each gene may participate in multiple pathways [12].

Several recent work [3, 12, 15, 18, 29] studies the overlapping group Lasso, where groups of features are given, potentially with overlaps between the groups. The resulting optimization is, however, much more challenging to solve due to the group overlaps. When optimizing the overlapping group Lasso problem, one can reformulate it as a second order cone program and solve it by a generic toolbox, which, however, does not scale well. Jenatton *et al*. [13] proposed an alternating algorithm called SLasso for solving the equivalent reformulation. However, SLasso involves an expensive matrix inversion at each alternating iteration, and there is no known global convergence rate for such an alternating procedure. A reformulation [5] was also proposed such that the original problem can be solved by the Alternating Direction Method of Multipliers (ADMM), which involves solving a linear system at each iteration, and may not scale well for high dimensional problems. Argyriou *et al*. [1] adopted the proximal gradient method for solving the overlapping group lasso, and a fixed point method was developed to compute the proximal operator. Chen *et al*. [6] employed a

smoothing technique to solve the overlapping group Lasso problem. Mairal [18] proposed to solve the proximal operator associated with the overlapping group Lasso defined as the sum of the ℓ_∞ norms, which, however, is not applicable to the overlapping group Lasso defined as the sum of the ℓ_2 norms considered in this paper.

In this paper, we develop an efficient algorithm for the overlapping group Lasso penalized problem via the accelerated gradient descent method. The accelerated gradient descent method has recently received increasing attention in machine learning due to the fast convergence rate even for non-smooth convex problems. One of the key operations is the computation of the proximal operator associated with the penalty. We reveal several key properties of the proximal operator associated with the overlapping group Lasso penalty, and proposed several possible reformulations that can be solved efficiently. The main contributions of this paper include: (1) we develop a low cost prepossessing procedure to identify (and then remove) zero groups in the proximal operator, which dramatically reduces the size of the problem to be solved; (2) we propose one dual formulation and two proximal splitting formulations for the proximal operator; (3) for the dual formulation, we further derive the duality gap which can be used to check the quality of the solution and determine the convergence of the algorithm. We have performed empirical evaluations using both synthetic data and the breast cancer gene expression data set, which consists of 8,141 genes organized into (overlapping) gene sets. Experimental results demonstrate the efficiency of the proposed algorithm in comparison with existing state-of-the-art algorithms.

Notations: $\|\cdot\|$ denotes the Euclidean norm, and $\mathbf{0}$ denotes a vector of zeros. $\mathrm{SGN}(\cdot)$ and $\mathrm{sgn}(\cdot)$ are defined in a component wise fashion as: 1) if $t = 0$, then $\mathrm{SGN}(t) = [-1, 1]$ and $\mathrm{sgn}(t) = 0$; 2) if $t > 0$, then $\mathrm{SGN}(t) = \{1\}$ and $\mathrm{sgn}(t) = 1$; and 3) if $t < 0$, $\mathrm{SGN}(t) = \{-1\}$ and $\mathrm{sgn}(t) = -1$. $G_i \subseteq \{1, 2, \ldots, p\}$ denotes an index set, and $\mathbf{x}_{G_i}$ denote a sub-vector of $\mathbf{x}$ restricted to G_i.

2 The Overlapping Group Lasso

We consider the following overlapping group Lasso penalized problem:

$$\min_{\mathbf{x}\in\mathbb{R}^p} f(\mathbf{x}) = l(\mathbf{x}) + \phi^{\lambda_1}_{\lambda_2}(\mathbf{x}) \tag{1}$$

where $l(\cdot)$ is a smooth convex loss function, e.g., the least squares loss,

$$\phi^{\lambda_1}_{\lambda_2}(\mathbf{x}) = \lambda_1\|\mathbf{x}\|_1 + \lambda_2 \sum_{i=1}^{g} w_i \|\mathbf{x}_{G_i}\| \tag{2}$$

is the overlapping group Lasso penalty, $\lambda_1 \geq 0$ and $\lambda_2 \geq 0$ are regularization parameters, $w_i > 0, i = 1, 2, \ldots, g$, $G_i \subseteq \{1, 2, \ldots, p\}$ contains the indices corresponding to the i-th group of features, and $\|\cdot\|$ denotes the Euclidean norm. Note that the first term in (2) can be absorbed into the second term, which however will introduce p additional groups. The g groups of features are pre-specified, and they may overlap. The penalty in (2) is a special case of the more general Composite Absolute Penalty (CAP) family [29]. When the groups are disjoint with $\lambda_1 = 0$ and $\lambda_2 > 0$, the model in (1) reduces to the group Lasso [28]. If $\lambda_1 > 0$ and $\lambda_2 = 0$, then the model in (1) reduces to the standard Lasso [25].

In this paper, we propose to make use of the accelerated gradient descent (AGD) [2, 21, 22] for solving (1), due to its fast convergence rate. The algorithm is called "FoGLasso", which stands for **F**ast **o**verlapping **G**roup **Lasso**. One of the key steps in the proposed FoGLasso algorithm is the computation of the proximal operator associated with the penalty in (2); and we present an efficient algorithm for the computation in the next section.

In FoGLasso, we first construct a model for approximating $f(\cdot)$ at the point $\mathbf{x}$ as:

$$f_{L,\mathbf{x}}(\mathbf{y}) = [l(\mathbf{x}) + \langle l'(\mathbf{x}), \mathbf{y} - \mathbf{x}\rangle] + \phi^{\lambda_2}_{\lambda_1}(\mathbf{y}) + \frac{L}{2}\|\mathbf{y} - \mathbf{x}\|^2, \tag{3}$$

where $L > 0$. The model $f_{L,\mathbf{x}}(\mathbf{y})$ consists of the first-order Taylor expansion of the smooth function $l(\cdot)$ at the point $\mathbf{x}$, the non-smooth penalty $\phi^{\lambda_2}_{\lambda_1}(\mathbf{x})$, and a regularization term $\frac{L}{2}\|\mathbf{y} - \mathbf{x}\|^2$. Next, a sequence of approximate solutions $\{\mathbf{x}_i\}$ is computed as follows: $\mathbf{x}_{i+1} = \arg\min_{\mathbf{y}} f_{L_i,\mathbf{s}_i}(\mathbf{y})$, where the search point $\mathbf{s}_i$ is an affine combination of $\mathbf{x}_{i-1}$ and $\mathbf{x}_i$ as $\mathbf{s}_i = \mathbf{x}_i + \beta_i(\mathbf{x}_i - \mathbf{x}_{i-1})$, for a properly chosen coefficient β_i, L_i is determined by the line search according to the Armijo-Goldstein rule

so that L_i should be appropriate for $\mathbf{s}_i$, i.e., $f(\mathbf{x}_{i+1}) \leq f_{L_i,\mathbf{s}_i}(\mathbf{x}_{i+1})$. A key building block in FoGLasso is the minimization of (3), whose solution is known as the proximal operator [20]. The computation of the proximal operator is the main technical contribution of this paper. The pseudo-code of FoGLasso is summarized in Algorithm 1, where the proximal operator $\pi(\cdot)$ is defined in (4). In practice, we can terminate Algorithm 1 if the change of the function values corresponding to adjacent iterations is within a small value, say 10^{-5}.

Algorithm 1 The FoGLasso Algorithm

Input: $L_0 > 0, \mathbf{x}_0, k$
Output: $\mathbf{x}_{k+1}$
1: Initialize $\mathbf{x}_1 = \mathbf{x}_0$, $\alpha_{-1} = 0$, $\alpha_0 = 1$, and $L = L_0$.
2: **for** $i = 1$ to k **do**
3: Set $\beta_i = \frac{\alpha_{i-2}-1}{\alpha_{i-1}}$, $\mathbf{s}_i = \mathbf{x}_i + \beta_i(\mathbf{x}_i - \mathbf{x}_{i-1})$
4: Find the smallest $L = 2^j L_{i-1}, j = 0, 1, \ldots$ such that $f(\mathbf{x}_{i+1}) \leq f_{L,\mathbf{s}_i}(\mathbf{x}_{i+1})$ holds, where $\mathbf{x}_{i+1} = \pi^{\lambda_1/L}_{\lambda_2/L}(\mathbf{s}_i - \frac{1}{L}l'(\mathbf{s}_i))$
5: Set $L_i = L$ and $\alpha_{i+1} = \frac{1+\sqrt{1+4\alpha_i^2}}{2}$
6: **end for**

3 The Associated Proximal Operator and Its Efficient Computation

The proximal operator associated with the overlapping group Lasso penalty is defined as follows:

$$\pi^{\lambda_1}_{\lambda_2}(\mathbf{v}) = \arg\min_{\mathbf{x}\in\mathbb{R}^p} \left\{ g^{\lambda_1}_{\lambda_2}(\mathbf{x}) \equiv \frac{1}{2}\|\mathbf{x}-\mathbf{v}\|^2 + \phi^{\lambda_1}_{\lambda_2}(\mathbf{x}) \right\}, \tag{4}$$

which is a special case of (1) by setting $l(\mathbf{x}) = \frac{1}{2}\|\mathbf{x}-\mathbf{v}\|^2$. It can be verified that the approximate solution $\mathbf{x}_{i+1} = \arg\min_{\mathbf{y}} f_{L_i,\mathbf{s}_i}(\mathbf{y})$ is given by $\mathbf{x}_{i+1} = \pi^{\lambda_1/L_i}_{\lambda_2/L_i}(\mathbf{s}_i - \frac{1}{L_i}l'(\mathbf{s}_i))$. Recently, it has been shown in [14] that, the efficient computation of the proximal operator is key to many sparse learning algorithms. Next, we focus on the efficient computation of $\pi^{\lambda_1}_{\lambda_2}(\mathbf{v})$ in (4) for a given $\mathbf{v}$. The rest of this section is organized as follows. In Section 3.1, we discuss some key properties of the proximal operator, based on which we propose a pre-processing technique that will significantly reduce the size of the problem. We then proposed to solve it via the dual formulation in Section 3.2, and the duality gap is also derived. Several alternative methods for solving the proximal operator via proximal splitting methods are discussed in Section 3.3.

3.1 Key Properties of the Proximal Operator

We first reveal several basic properties of the proximal operator $\pi^{\lambda_1}_{\lambda_2}(\mathbf{v})$.

Lemma 1. *Suppose that $\lambda_1, \lambda_2 \geq 0$, and $w_i > 0$, for $i = 1, 2, \ldots, g$. Let $\mathbf{x}^* = \pi^{\lambda_1}_{\lambda_2}(\mathbf{v})$. The following holds: 1) if $v_i > 0$, then $0 \leq x^*_i \leq v_i$; 2) if $v_i < 0$, then $v_i \leq x^*_i \leq 0$; 3) if $v_i = 0$, then $x^*_i = 0$; 4) $\mathrm{SGN}(\mathbf{v}) \subseteq \mathrm{SGN}(\mathbf{x}^*)$; and 5) $\pi^{\lambda_1}_{\lambda_2}(\mathbf{v}) = \mathrm{sgn}(\mathbf{v}) \odot \pi^{\lambda_1}_{\lambda_2}(|\mathbf{v}|)$.*

Proof. When $\lambda_1, \lambda_2 \geq 0$, and $w_i \geq 0$, for $i = 1, 2, \ldots, g$, the objective function $g^{\lambda_1}_{\lambda_2}(\cdot)$ is strictly convex, thus $\mathbf{x}^*$ is the unique minimizer. We first show if $v_i > 0$, then $0 \leq x^*_i \leq v_i$. If $x^*_i > v_i$, then we can construct a $\hat{\mathbf{x}}$ as follows: $\hat{x}_j = x^*_j, j \neq i$ and $\hat{x}_i = v_i$. Similarly, if $x^*_i < 0$, then we can construct a $\hat{\mathbf{x}}$ as follows: $\hat{x}_j = x^*_j, j \neq i$ and $\hat{x}_i = 0$. It is easy to verify that $\hat{\mathbf{x}}$ achieves a lower objective function value than $\mathbf{x}^*$ in both cases. We can prove the second and the third properties using similar arguments. Finally, we can prove the fourth and the fifth properties using the definition of $\mathrm{SGN}(\cdot)$ and the first three properties. □

Next, we show that $\pi^{\lambda_1}_{\lambda_2}(\cdot)$ can be directly derived from $\pi^0_{\lambda_2}(\cdot)$ by soft-thresholding. Thus, we only need to focus on the case when $\lambda_1 = 0$. This simplifies the optimization in (4). It is an extension of the result for Fused Lasso in [10].

Theorem 1. *Let $\mathbf{u} = \mathrm{sgn}(\mathbf{v}) \odot \max(|\mathbf{v}| - \lambda_1, 0)$, and*

$$\pi^0_{\lambda_2}(\mathbf{u}) = \arg\min_{\mathbf{x}\in\mathbb{R}^p} \left\{ h_{\lambda_2}(\mathbf{x}) \equiv \frac{1}{2}\|\mathbf{x}-\mathbf{u}\|^2 + \lambda_2 \sum_{i=1}^{g} w_i \|\mathbf{x}_{G_i}\| \right\}. \tag{5}$$

Then, the following holds: $\pi^{\lambda_1}_{\lambda_2}(\mathbf{v}) = \pi^0_{\lambda_2}(\mathbf{u})$.

Proof. Denote the unique minimizer of $h_{\lambda_2}(\cdot)$ as $\mathbf{x}^*$. The sufficient and necessary condition for the optimality of $\mathbf{x}^*$ is:

$$\mathbf{0} \in \partial h_{\lambda_2}(\mathbf{x}^*) = \mathbf{x}^* - \mathbf{u} + \partial \phi^0_{\lambda_2}(\mathbf{x}^*), \tag{6}$$

where $\partial h_{\lambda_2}(\mathbf{x})$ and $\partial \phi^0_{\lambda_2}(\mathbf{x})$ are the sub-differential sets of $h_{\lambda_2}(\cdot)$ and $\phi^0_{\lambda_2}(\cdot)$ at $\mathbf{x}$, respectively.

Next, we need to show $\mathbf{0} \in \partial g^{\lambda_1}_{\lambda_2}(\mathbf{x}^*)$. The sub-differential of $g^{\lambda_1}_{\lambda_2}(\cdot)$ at $\mathbf{x}^*$ is given by

$$\partial g^{\lambda_1}_{\lambda_2}(\mathbf{x}^*) = \mathbf{x}^* - \mathbf{v} + \partial \phi^{\lambda_1}_{\lambda_2}(\mathbf{x}^*) = \mathbf{x}^* - \mathbf{v} + \lambda_1 \mathrm{SGN}(\mathbf{x}^*) + \partial \phi^0_{\lambda_2}(\mathbf{x}^*). \tag{7}$$

It follows from the definition of $\mathbf{u}$ that $\mathbf{u} \in \mathbf{v} - \lambda_1 \mathrm{SGN}(\mathbf{u})$. Using the fourth property in Lemma 1, we have $\mathrm{SGN}(\mathbf{u}) \subseteq \mathrm{SGN}(\mathbf{x}^*)$. Thus,

$$\mathbf{u} \in \mathbf{v} - \lambda_1 \mathrm{SGN}(\mathbf{x}^*). \tag{8}$$

It follows from (6)-(8) that $\mathbf{0} \in \partial g^{\lambda_1}_{\lambda_2}(\mathbf{x}^*)$. □

It follows from Theorem 1 that, we only need to focus on the optimization of (5) in the following discussion. The difficulty in the optimization of (5) lies in the large number of groups that may overlap. In practice, many groups will be zero, thus achieving a sparse solution (a sparse solution is desirable in many applications). However, the zero groups are not known in advance. The key question we aim to address is how we can identify as many zero groups as possible to reduce the complexity of the optimization. Next, we present a sufficient condition for a group to be zero.

Lemma 2. *Denote the minimizer of $h_{\lambda_2}(\cdot)$ in* (5) *by $\mathbf{x}^*$. If the i-th group satisfies $\|\mathbf{u}_{G_i}\| \leq \lambda_2 w_i$, then $\mathbf{x}^*_{G_i} = \mathbf{0}$, i.e., the i-th group is zero.*

Proof. We decompose $h_{\lambda_2}(\mathbf{x})$ into two parts as follows:

$$h_{\lambda_2}(\mathbf{x}) = \left(\frac{1}{2}\|\mathbf{x}_{G_i} - \mathbf{u}_{G_i}\|^2 + \lambda_2 w_i \|\mathbf{x}_{G_i}\|\right) + \left(\frac{1}{2}\|\mathbf{x}_{\overline{G}_i} - \mathbf{u}_{\overline{G}_i}\|^2 + \lambda_2 \sum_{j \neq i} w_j \|\mathbf{x}_{G_j}\|\right), \tag{9}$$

where $\overline{G}_i = \{1, 2, \ldots, p\} - G_i$ is the complementary set of G_i. We consider the minimization of $h_{\lambda_2}(\mathbf{x})$ in terms of $\mathbf{x}_{G_i}$ when $\mathbf{x}_{\overline{G}_i} = \mathbf{x}^*_{\overline{G}_i}$ is fixed. It can be verified that if $\|\mathbf{u}_{G_i}\| \leq \lambda_2 w_i$, then $\mathbf{x}^*_{G_i} = \mathbf{0}$ minimizes both terms in (9) simultaneously. Thus we have $\mathbf{x}^*_{G_i} = \mathbf{0}$. □

Lemma 2 may not identify many true zero groups due to the strong condition imposed. The lemma below weakens the condition in Lemma 2. Intuitively, for a group G_i, we first identify all existing zero groups that overlap with G_i, and then compute the overlapping index subset S_i of G_i as:

$$S_i = \bigcup_{j \neq i, \mathbf{x}^*_{G_j} = \mathbf{0}} (G_j \cap G_i). \tag{10}$$

We can show that $\mathbf{x}^*_{G_i} = \mathbf{0}$ if $\|\mathbf{u}_{G_i - S_i}\| \leq \lambda_2 w_i$ is satisfied. Note that this condition is much weaker than the condition in Lemma 2, which requires that $\|\mathbf{u}_{G_i}\| \leq \lambda_2 w_i$.

Lemma 3. *Denote the minimizer of $h_{\lambda_2}(\cdot)$ by $\mathbf{x}^*$. Let S_i, a subset of G_i, be defined in* (10)*. If $\|\mathbf{u}_{G_i - S_i}\| \leq \lambda_2 w_i$ holds, then $\mathbf{x}^*_{G_i} = \mathbf{0}$.*

Proof. Suppose that we have identified a collection of zero groups. By removing these groups, the original problem (5) can then be reduced to:

$$\min_{\mathbf{x}(I_1) \in \mathbb{R}^{|I_1|}} \frac{1}{2}\|\mathbf{x}(I_1) - \mathbf{u}(I_1)\|^2 + \lambda_2 \sum_{i \in \mathcal{G}_1} w_i \|\mathbf{x}_{G_i - S_i}\|$$

where I_1 is the reduced index set, i.e., $I_1 = \{1, 2, \ldots, p\} - \bigcup_{i: \mathbf{x}^*_{G_i} = \mathbf{0}} G_i$, and $\mathcal{G}_1 = \{i : \mathbf{x}^*_{G_i} \neq \mathbf{0}\}$ is the index set of the remaining non-zero groups. Note that $\forall i \in \mathcal{G}_1$, $G_i - S_i \in I_1$. By applying Lemma 2 again, we show that if $\|\mathbf{u}_{G_i - S_i}\| \leq \lambda_2 w_i$ holds, then $\mathbf{x}^*_{G_i - S_i} = \mathbf{0}$. Thus, $\mathbf{x}^*_{G_i} = \mathbf{0}$. □

Lemma 3 naturally leads to an iterative procedure for identifying the zero groups: For each group G_i, if $\|\mathbf{u}_{G_i}\| \leq \lambda_2 w_i$, then we set $\mathbf{u}_{G_i} = \mathbf{0}$; we cycle through all groups repeatedly until $\mathbf{u}$ does not change. Let $p' = |\{u_i : u_i \neq 0\}|$ be the number of nonzero elements in $\mathbf{u}$, $g' = |\{\mathbf{u}_{G_i} : \mathbf{u}_{G_i} \neq \mathbf{0}\}|$ be the number of the nonzero groups, and $\mathbf{x}^*$ denote the minimizer of $h_{\lambda_2}(\cdot)$. It follows from Lemma 3 and Lemma 1 that, if $u_i = 0$, then $x^*_i = 0$. Therefore, by applying the above iterative procedure, we can find the minimizer of (5) by solving a reduced problem that has $p' \leq p$ variables and $g' \leq g$ groups. With some abuse of notation, we still use (5) to denote the resulting reduced problem. In addition, from Lemma 1, we only focus on $\mathbf{u} > 0$ in the following discussion, and the analysis can be easily generalized to the general $\mathbf{u}$.

3.2 Reformulation as an Equivalent Smooth Convex Optimization Problem

It follows from the first two properties of Lemma 1 that, we can rewrite (5) as:

$$\pi^0_{\lambda_2}(\mathbf{u}) = \arg \min_{\substack{\mathbf{x}\in\mathbb{R}^p \\ \mathbf{0}\preceq\mathbf{x}\preceq\mathbf{u}}} h_{\lambda_2}(\mathbf{x}), \tag{11}$$

where $\preceq$ denotes the element-wise inequality, and

$$h_{\lambda_2}(\mathbf{x}) = \frac{1}{2}\|\mathbf{x}-\mathbf{u}\|^2 + \lambda_2 \sum_{i=1}^{g} w_i \|\mathbf{x}_{G_i}\|,$$

and the minimizer of $h_{\lambda_2}(\cdot)$ is constrained to be non-negative due to $\mathbf{u} > 0$ (refer to the discussion at the end of Section 3.1).

Making use of the dual norm of the Euclidean norm $\|\cdot\|$, we can rewrite $h_{\lambda_2}(\mathbf{x})$ as:

$$h_{\lambda_2}(\mathbf{x}) = \max_{Y\in\Omega} \frac{1}{2}\|\mathbf{x}-\mathbf{u}\|^2 + \sum_{i=1}^{g} \langle \mathbf{x}, Y^i\rangle, \tag{12}$$

where Ω is defined as follows:

$$\Omega = \{Y \in \mathbb{R}^{p\times g} : Y^i_{\overline{G}_i} = \mathbf{0}, \|Y^i\| \le \lambda_2 w_i, i = 1, 2, \ldots, g\},$$

$\overline{G}_i$ is the complementary set of G_i, Y is a sparse matrix satisfying $Y_{ij} = 0$ if the i-th feature does not belong to the j-th group, i.e., $i \notin G_j$, and Y^i denotes the i-th column of Y. As a result, we can reformulate (11) as the following min-max problem:

$$\min_{\substack{\mathbf{x}\in\mathbb{R}^p \\ \mathbf{0}\preceq\mathbf{x}\preceq\mathbf{u}}} \max_{Y\in\Omega} \left\{ \psi(\mathbf{x}, Y) = \frac{1}{2}\|\mathbf{x}-\mathbf{u}\|^2 + \langle \mathbf{x}, Y\mathbf{e}\rangle \right\}, \tag{13}$$

where $\mathbf{e} \in \mathbb{R}^g$ is a vector of ones. It is easy to verify that $\psi(\mathbf{x}, Y)$ is convex in $\mathbf{x}$ and concave in Y, and the constraint sets are closed convex for both $\mathbf{x}$ and Y. Thus, (13) has a saddle point, and the min-max can be exchanged.

It is easy to verify that for a given Y, the optimal $\mathbf{x}$ minimizing $\psi(\mathbf{x}, Y)$ in (13) is given by

$$\mathbf{x} = \max(\mathbf{u} - Y\mathbf{e}, \mathbf{0}). \tag{14}$$

Plugging (14) into (13), we obtain the following minimization problem with regard to Y:

$$\min_{Y\in\mathbb{R}^{p\times g}: Y\in\Omega} \{\omega(Y) = -\psi(\max(\mathbf{u} - Y\mathbf{e}, \mathbf{0}), Y)\}. \tag{15}$$

Our methodology for minimizing $h_{\lambda_2}(\cdot)$ defined in (5) is to first solve (15), and then construct the solution to $h_{\lambda_2}(\cdot)$ via (14). Using standard optimization techniques, we can show that the function $\omega(\cdot)$ is continuously differentiable with Lipschitz continuous gradient. We include the detailed proof in the supplemental material for completeness. Therefore, we convert the non-smooth problem (11) to the smooth problem (15), making the smooth convex optimization tools applicable. In this paper, we employ the accelerated gradient descent to solve (15), due to its fast convergence property. Note that, the Euclidean projection onto the set Ω can be computed in closed form. We would like to emphasize here that, the problem (15) may have a much smaller size than (4).

3.2.1 Computing the Duality Gap

We show how to estimate the duality gap of the min-max problem (13), which can be used to check the quality of the solution and determine the convergence of the algorithm.

For any given approximate solution $\tilde{Y} \in \Omega$ for $\omega(Y)$, we can construct the approximate solution $\tilde{\mathbf{x}} = \max(\mathbf{u} - \tilde{Y}\mathbf{e}, \mathbf{0})$ for $h_{\lambda_2}(\mathbf{x})$. The duality gap for the min-max problem (13) at the point $(\tilde{\mathbf{x}}, \tilde{Y})$ can be computed as:

$$\text{gap}(\tilde{Y}) = \max_{Y\in\Omega} \psi(\tilde{\mathbf{x}}, Y) - \min_{\substack{\mathbf{x}\in\mathbb{R}^p \\ \mathbf{0}\preceq\mathbf{x}\preceq\mathbf{u}}} \psi(\mathbf{x}, \tilde{Y}). \tag{16}$$

The main result of this subsection is summarized in the following theorem:

Theorem 2. *Let* $\mathrm{gap}(\tilde{Y})$ *be the duality gap defined in* (16)*. Then, the following holds:*

$$\mathrm{gap}(\tilde{Y}) = \lambda_2 \sum_{i=1}^{g} (w_i \|\tilde{\mathbf{x}}_{G_i}\| - \langle \tilde{\mathbf{x}}_{G_i}, \tilde{Y}^i_{G_i} \rangle). \tag{17}$$

In addition, we have

$$\omega(\tilde{Y}) - \omega(Y^*) \leq \mathrm{gap}(\tilde{Y}), \tag{18}$$

$$h(\tilde{\mathbf{x}}) - h(\mathbf{x}^*) \leq \mathrm{gap}(\tilde{Y}). \tag{19}$$

Proof. Denote $(\mathbf{x}^*, Y^*)$ as the optimal solution to the problem (13). From (12)-(15), we have

$$-\omega(\tilde{Y}) = \psi(\tilde{\mathbf{x}}, \tilde{Y}) = \min_{\substack{\mathbf{x}\in\mathbb{R}^p \\ \mathbf{0}\preceq\mathbf{x}\preceq\mathbf{u}}} \psi(\mathbf{x}, \tilde{Y}) \leq \psi(\mathbf{x}^*, \tilde{Y}), \tag{20}$$

$$\psi(\mathbf{x}^*, \tilde{Y}) \leq \max_{Y\in\Omega} \psi(\mathbf{x}^*, Y) = \psi(\mathbf{x}^*, Y^*) = -\omega(Y^*), \tag{21}$$

$$h_{\lambda_2}(\mathbf{x}^*) = \psi(\mathbf{x}^*, Y^*) = \min_{\substack{\mathbf{x}\in\mathbb{R}^p \\ \mathbf{0}\preceq\mathbf{x}\preceq\mathbf{u}}} \psi(\mathbf{x}, Y^*) \leq \psi(\tilde{\mathbf{x}}, Y^*), \tag{22}$$

$$\psi(\tilde{\mathbf{x}}, Y^*) \leq \max_{Y\in\Omega} \psi(\tilde{\mathbf{x}}, Y) = h_{\lambda_2}(\tilde{\mathbf{x}}). \tag{23}$$

Incorporating (11), (20)-(23), we prove (17)-(19). □

In our experiments, we terminate the algorithm when the estimated duality gap is less than 10^{-10}.

3.3 Proximal Splitting Methods

Recently, a family of proximal splitting methods [8] have been proposed for converting a challenging optimization problem into a series of sub-problems with a closed form solution. We consider two reformulations of the proximal operator (4), based on the Dykstra-like Proximal Splitting Method and the alternating direction method of multipliers (ADMM). The efficiency of these two methods for overlapping Group Lasso will be demonstrated in the next section.

In [5], Boyd *et al.* suggested that the original overlapping group lasso problem (1) can be reformulated and solved by ADMM directly. We include the implementation of ADMM for our comparative study. We provide the details of all three reformulations in the supplemental materials.

4 Experiments

In this section, extensive experiments are performed to demonstrate the efficiency of our proposed methods. We use both synthetic data sets and a real world data set and the evaluation is done in various problem size and precision settings. The proposed algorithms are mainly implemented in Matlab, with the proximal operator implemented in standard C for improved efficiency. Several state-of-the-art methods are also included for comparison purpose, including SLasso algorithm developed by Jenatton *et al*. [13], the ADMM reformulation [5], the Prox-Grad method by Chen *et al*. [6] and the Picard-Nesterov algorithm by Argyriou *et al*. [1].

4.1 Synthetic Data

In the first set of simulation we consider only the key component of our algorithm, the proximal operator. The group indices are predefined such that $G_1 = \{1, 2, \ldots, 10\}$, $G_2 = \{6, 7, \ldots, 20\}, \ldots$, with each group overlapping half of the previous group. 100 examples are generated for each set of fixed problem size p and group size g, and the results are summarized in Figure 1. As we can observe from the figure, the dual formulation yields the best performance, followed closely by ADMM and then the Dykstra method. We can also observe that our method scales very well to high dimensional problems, since even with $p = 10^6$, the proximal operator can be computed in a few seconds. It is also not surprising that Dykstra method is much more sensitive to the number of groups, which equals to the number of projections in one Dykstra step.

To illustrate the effectiveness of our pre-processing technique, we repeat the previous experiment by removing the pre-processing step. The results are shown in the right plot of Figure 1. As we can observe from the figure, the proposed pre-processing technique effectively reduces the computational

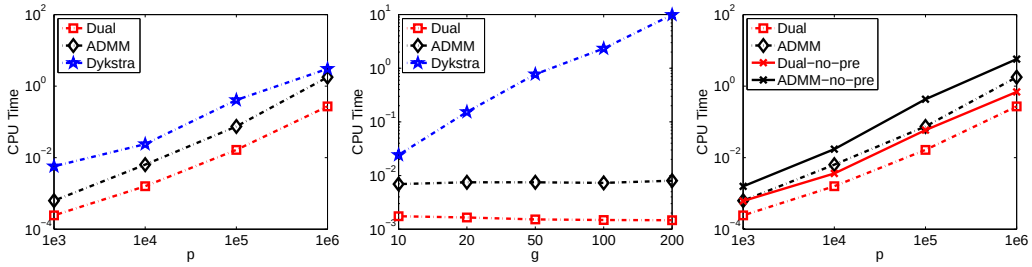

Figure 1: Time comparison for computing the proximal operators. The group number is fixed in the left figure and the problem size is fixed in the middle figure. In the right figure, the effectiveness of the pre-processing technique is illustrated.

time. As is evident from Figure 1, the dual formulation proposed in Section 3.2 consistently outperforms other proximal splitting methods. In the following experiments, only the dual method will be used for computing the proximal operator, and our method will then be called as "FoGLasso".

4.2 Gene Expression Data

We have also conducted experiments to evaluate the efficiency of the proposed algorithm using the breast cancer gene expression data set [26], which consists of 8,141 genes in 295 breast cancer tumors (78 metastatic and 217 non-metastatic). For the sake of analyzing microarrays in terms of biologically meaningful gene sets, different approaches have been used to organize the genes into (overlapping) gene sets. In our experiments, we follow [12] and employ the following two approaches for generating the overlapping gene sets (groups): pathways [24] and edges [7]. For pathways, the canonical pathways from the Molecular Signatures Database (MSigDB) [24] are used. It contains 639 groups of genes, of which 637 groups involve the genes in our study. The statistics of the 637 gene groups are summarized as follows: the average number of genes in each group is 23.7, the largest gene group has 213 genes, and 3,510 genes appear in these 637 groups with an average appearance frequency of about 4. For edges, the network built in [7] will be used, and we follow [12] to extract 42,594 edges from the network, leading to 42,594 overlapping gene sets of size 2. All 8,141 genes appear in the 42,594 groups with an average appearance frequency of about 10. The experimental settings are as follows: we solve (1) with the least squares loss $l(\mathbf{x}) = \frac{1}{2}\|A\mathbf{x} - \mathbf{b}\|^2$, and we set $w_i = \sqrt{|G_i|}$, and $\lambda_1 = \lambda_2 = \gamma \times \lambda_1^{\max}$, where $|G_i|$ denotes the size of the i-th group G_i, $\lambda_1^{\max} = \|A^{\mathrm{T}}\mathbf{b}\|_\infty$ (the zero point is a solution to (1) if $\lambda_1 \geq \lambda_1^{\max}$), and γ is chosen from the set $\{5\times10^{-1}, 2\times10^{-1}, 1\times10^{-1}, 5\times10^{-2}, 2\times10^{-2}, 1\times10^{-2}, 5\times10^{-3}, 2\times10^{-3}, 1\times10^{-3}\}$.

Comparison with SLasso, Prox-Grad and ADMM We first compare our proposed FoGLasso with the SLasso algorithm [13], ADMM [5] and Prox-Grad [6]. The comparisons are based on the computational time, since all these methods have efficient Matlab implementations with key components written in C. For a given γ, we first run SLasso till a certain precision level is reached, and then run the others until they achieve an objective function value smaller than or equal to that of SLasso. Different precision levels of the solutions are evaluated such that a fair comparison can be made. We vary the number of genes involved, and report the total computational time (seconds) including all nine regularization parameters in Figure 2. We can observe that: 1) for all precision levels, our proposed FoGLasso is much more efficient than SLasso, ADMM and Prox-Grad; 2) the advantage of FoGLasso over other three methods in efficiency grows with the increasing number of genes (variables). For example, with the grouping by pathways, FoGLasso is about 25 and 70 times faster than SLasso for 1000 and 2000 genes, respectively; and 3) the efficiency on edges is inferior to that on pathways, due to the larger number of overlapping groups. Additional scalability study of our proposed method using larger problem size can be found in the supplemental materials.

Comparison with Picard-Nesterov Since the code acquired for Picard-Nesterov is implemented purely in Matlab, a computational time comparison might not be fair. Therefore, only the number of iterations required for convergence is reported, as both methods adopt the first order method. We use edges to generate the groups, and vary the problem size from 100 to 400, using the same set of regularization parameters. For each problem, we record both the number of outer iterations (the gradient steps) and the total number of inner iterations (the steps required for computing the

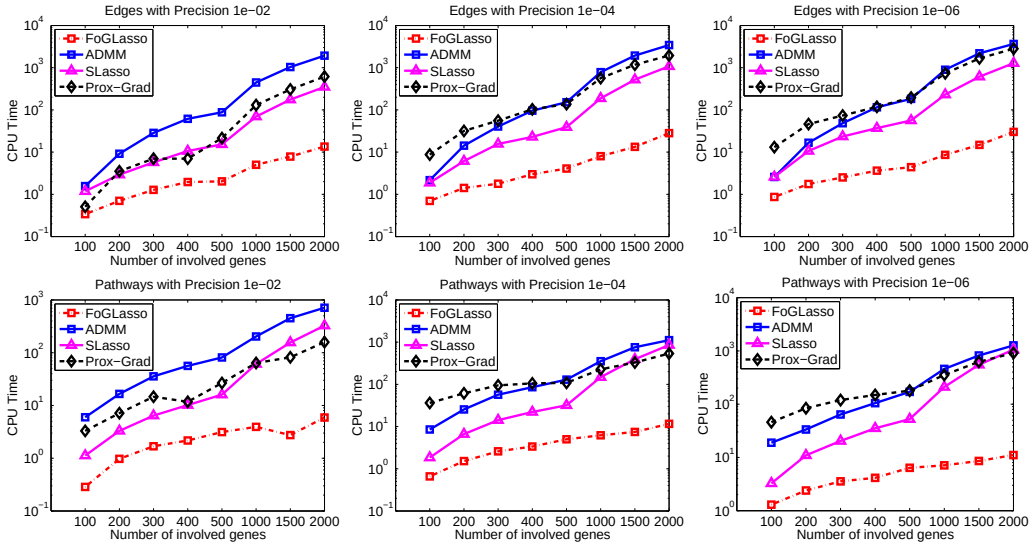

Figure 2: Comparison of SLasso [13], ADMM [5], Prox-Grad [6] and our proposed FoGLasso algorithm in terms of computational time (in seconds and in the logarithmic scale) when different numbers of genes (variables) are involved. Different precision levels are used for comparison.

Table 1: Comparison of FoGLasso and Picard-Nesterov using different numbers (p) of genes and various precision levels. For each particular method, the first row denotes the number of outer iterations required for convergence, while the second row represents the total number of inner iterations.

Precision Level	10^{-2}			10^{-4}			10^{-6}		
p	100	200	400	100	200	400	100	200	400
FoGLasso	81	189	353	192	371	1299	334	507	1796
	288	401	921	404	590	1912	547	727	2387
Picard-Nesterov	78	176	325	181	304	1028	318	504	1431
	8271	6.8e4	2.2e5	2.6e4	1.0e5	7.8e5	5.1e4	1.3e5	1.1e6

proximal operators). The average number of iterations among all the regularization parameters are summarized in Table 1. As we can observe from the table, though Picard-Nesterov method often takes less outer iterations to converge, it takes a lot more inner iterations to compute the proximal operator. It is straight forward to verify that the inner iterations in Picard-Nesterov method and our proposed method have the same complexity of $O(pg)$.

5 Conclusion

In this paper, we consider the efficient optimization of the overlapping group Lasso penalized problem based on the accelerated gradient descent method. We reveal several key properties of the proximal operator associated with the overlapping group Lasso, and compute the proximal operator via solving the smooth and convex dual problem. Numerical experiments on both synthetic and the breast cancer data set demonstrate the efficiency of the proposed algorithm. Although with an inexact proximal operator, the optimal convergence rate of the accelerated gradient descent might not be guaranteed [23, 11], the algorithm performs quite well empirically. A theoretical analysis on the convergence property will be an interesting future direction. In the future, we also plan to apply the proposed algorithm to other real-world applications involving overlapping groups.

Acknowledgments

This work was supported by NSF IIS-0812551, IIS-0953662, MCB-1026710, CCF-1025177, NGA HM1582-08-1-0016, and NSFC 60905035, 61035003.

References

[1] A. Argyriou, C.A. Micchelli, M. Pontil, L. Shen, and Y. Xu. Efficient first order methods for linear composite regularizers. *Arxiv preprint arXiv:1104.1436*, 2011.

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

[3] H. D. Bondell and B. J. Reich. Simultaneous regression shrinkage, variable selection and clustering of predictors with oscar. *Biometrics*, 64:115–123, 2008.

[4] J. F. Bonnans and A. Shapiro. Optimization problems with perturbations: A guided tour. *SIAM Review*, 40(2):228–264, 1998.

[5] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. 2010.

[6] X. Chen, Q. Lin, S. Kim, J.G. Carbonell, and E.P. Xing. An efficient proximal gradient method for general structured sparse learning. *Arxiv preprint arXiv:1005.4717*, 2010.

[7] H. Y. Chuang, E. Lee, Y. T. Liu, D. Lee, and T. Ideker. Network-based classification of breast cancer metastasis. *Molecular Systems Biology*, 3(140), 2007.

[8] P.L. Combettes and J.C. Pesquet. Proximal splitting methods in signal processing. *Arxiv preprint arXiv:0912.3522*, 2009.

[9] J. M. Danskin. *The theory of max-min and its applications to weapons allocation problems*. Springer-Verlag, New York, 1967.

[10] J. Friedman, T. Hastie, H. Höfling, and R. Tibshirani. Pathwise coordinate optimization. *Annals of Applied Statistics*, 1(2):302–332, 2007.

[11] B. He and X. Yuan. An accelerated inexact proximal point algorithm for convex minimization. 2010.

[12] L. Jacob, G. Obozinski, and J. Vert. Group lasso with overlap and graph lasso. In *ICML*, 2009.

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

[14] R. Jenatton, J. Mairal, G. Obozinski, and F. Bach. Proximal methods for sparse hierarchical dictionary learning. In *ICML*, 2010.

[15] S. Kim and E. P. Xing. Tree-guided group lasso for multi-task regression with structured sparsity. In *ICML*, 2010.

[16] H. Liu, M. Palatucci, and J. Zhang. Blockwise coordinate descent procedures for the multi-task lasso, with applications to neural semantic basis discovery. In *ICML*, 2009.

[17] J. Liu, S. Ji, and J. Ye. Multi-task feature learning via efficient $\ell_{2,1}$-norm minimization. In *UAI*, 2009.

[18] J. Mairal, R. Jenatton, G. Obozinski, and F. Bach. Network flow algorithms for structured sparsity. In *NIPS*. 2010.

[19] L. Meier, S. Geer, and P. Bühlmann. The group lasso for logistic regression. *Journal of the Royal Statistical Society: Series B*, 70:53–71, 2008.

[20] J.-J. Moreau. Proximité et dualité dans un espace hilbertien. *Bull. Soc. Math. France*, 93:273–299, 1965.

[21] A. Nemirovski. *Efficient methods in convex programming*. Lecture Notes, 1994.

[22] Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Kluwer Academic Publishers, 2004.

[23] R.T. Rockafellar. Monotone operators and the proximal point algorithm. *SIAM Journal on Control and Optimization*, 14:877, 1976.

[24] A. Subramanian and et al. Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences*, 102(43):15545–15550, 2005.

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

[26] M. J. Van de Vijver and et al. A gene-expression signature as a predictor of survival in breast cancer. *The New England Journal of Medicine*, 347(25):1999–2009, 2002.

[27] Y. Ying, C. Campbell, and M. Girolami. Analysis of svm with indefinite kernels. In *NIPS*. 2009.

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

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

Efficient Methods for Overlapping Group Lasso: Supplemental Material

A. Properties of the Function $\omega(\cdot)$ in (15)

Theorem 3. *The function $\omega(Y)$ is convex and continuously differentiable with*

$$\omega'(Y) = -\max(\mathbf{u} - Y\mathbf{e}, \mathbf{0})\mathbf{e}^{\mathrm{T}}. \tag{24}$$

In addition, $\omega'(Y)$ is Lipschitz continuous with constant g^2, i.e.,

$$\|\omega'(Y_1) - \omega'(Y_2)\|_F \leq g^2\|Y_1 - Y_2\|_F, \quad \forall\ Y_1, Y_2 \in \mathbb{R}^{p\times g}. \tag{25}$$

To prove Theorem 3, we first present two technical lemmas. The first lemma is related to the optimal value function [4, 9], and it was used in a recent study [27] on infinite kernel learning.

Lemma 4. *[4] Let X be a metric space and U be a normed space. Suppose that for all $\mathbf{x} \in X$, the function $\psi(\mathbf{x}, \cdot)$ is differentiable and that $\psi(\mathbf{x}, Y)$ and $D_Y\psi(\mathbf{x}, Y)$ (the partial derivative of $\psi(\mathbf{x}, Y)$ with respect to Y) are continuous on $X \times U$. Let Φ be a compact subset of X. Define the optimal value function as $\varphi(Y) = \inf_{\mathbf{x}\in\Phi} \psi(\mathbf{x}, Y)$. The optimal value function $\varphi(Y)$ is directionally differentiable. In addition, if $\forall Y \in U$, $\psi(\cdot, Y)$ has a unique minimizer $\mathbf{x}(Y)$ over Φ, then $\varphi(Y)$ is differentiable at Y and the gradient of $\varphi(Y)$ is given by $\varphi'(Y) = D_Y\psi(\mathbf{x}(Y), Y)$.*

The second lemma shows that the operator $\mathbf{y} = \max(\mathbf{x}, \mathbf{0})$ is non-expansive.

Lemma 5. *$\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^p$, we have $\|\max(\mathbf{x}, \mathbf{0}) - \max(\mathbf{y}, \mathbf{0})\| \leq \|\mathbf{x} - \mathbf{y}\|$.*

Proof. The results follows since $|\max(x, 0) - \max(y, 0)| \leq |x - y|$, $\forall x, y \in \mathbb{R}$. □

Proof of Theorem 3: To prove the differentiability of $\omega(Y)$, we apply Lemma 4 with $X = \mathbb{R}^p$, $U = \mathbb{R}^{p\times g}$ and $\Phi = \{\mathbf{x} \in X : \mathbf{u} + \lambda_2 \sum w_i\mathbf{e} \geq \mathbf{x} \geq \mathbf{0}\}$. It is easy to verify that 1) $\psi(\mathbf{x}, \cdot)$ is differentiable; 2) $\psi(\mathbf{x}, Y)$ and $D_Y\psi(\mathbf{x}, Y) = \mathbf{x}\mathbf{e}^{\mathrm{T}}$ are continuous on $X \times U$; 3) Φ be a compact subset of X; and 4) $\forall Y \in U$, $\psi(\mathbf{x}, Y)$ has a unique minimizer $\mathbf{x}(Y) = \max(\mathbf{u} - Y\mathbf{e}, \mathbf{0})$ over Φ. Note that, the last result follows from $\mathbf{u} > 0$ and $\mathbf{u} - Y\mathbf{e} \leq \mathbf{u} + \lambda_2 \sum w_i\mathbf{e}$, where the latter inequality utilizes $\|Y^i\| \leq \lambda_2 w_i$; and this indicates that $\mathbf{x}(Y) = \max(\mathbf{u} - Y\mathbf{e}, \mathbf{0}) = \arg\min_{\mathbf{x}} \psi(\mathbf{x}, Y) = \arg\min_{\mathbf{x}\in\Phi} \psi(\mathbf{x}, Y)$. It follows from Lemma 4 that

$$\varphi(Y) = \inf_{\mathbf{x}\in\Phi} \psi(\mathbf{x}, Y) = \psi(\max(\mathbf{u} - Y\mathbf{e}, \mathbf{0}), Y)$$

is differentiable with $\varphi'(Y) = \max(\mathbf{u} - Y\mathbf{e}, \mathbf{0})\mathbf{e}^{\mathrm{T}}$.

In (13), $\psi(\mathbf{x}, Y)$ is convex in $\mathbf{x}$ and concave in Y, and the constraint sets are closed convex for both $\mathbf{x}$ and Y, thus the existence of the saddle point is guaranteed by the well-known von Neumann Lemma [21]. As a result,

$$\varphi(Y) = \inf_{\mathbf{x}\in\Phi} \psi(\mathbf{x}, Y) = \psi(\max(\mathbf{u} - Y\mathbf{e}, \mathbf{0}), Y)$$

is concave, and $\omega(Y) = -\varphi(Y)$ is convex. For any Y_1, Y_2, we have

$$\begin{aligned}
&\|\omega'(Y_1) - \omega'(Y_2)\|_F \\
=&\|\max(\mathbf{u} - Y_1\mathbf{e}, \mathbf{0})\mathbf{e}^{\mathrm{T}} - \max(\mathbf{u} - Y_2\mathbf{e}, \mathbf{0})\mathbf{e}^{\mathrm{T}}\|_F \\
\leq&\|\mathbf{e}\| \times \|\max(\mathbf{u} - Y_1\mathbf{e}, \mathbf{0}) - \max(\mathbf{u} - Y_2\mathbf{e}, \mathbf{0})\| \\
\leq&\|\mathbf{e}\| \times \|(Y_1 - Y_2)\mathbf{e}\| \\
\leq&g^2\|Y_1 - Y_2\|_F,
\end{aligned} \tag{26}$$

where the second inequality follows from Lemma 5. We prove (25). □

B. Dykstra-like Proximal Splitting Method for Computing the Proximal Operator in (5)

In the field of signal processing, one classical problem is the *convex feasibility problem*:

$$\text{find } x \in \bigcap_{i=1}^{m} C_i, \tag{27}$$

where C_i's are convex sets. Efficient methods have been designed for (27) where at each iteration, only one convex set is considered and the solution is updated iteratively by cycling through all convex sets. Under certain conditions, convergence is guaranteed. For our problem, since (5) can be considered as the projection of a vector $\mathbf{u}$ onto a collection of convex sets induced by the regularization components $w_i\|\mathbf{x}_{G_i}\|$, the proximal splitting ideas can be applied.

We define $f_i = \lambda\|\mathbf{x}_{G_i}\|$, the proximal operator in (5) can be rewritten as:

$$\min_{\mathbf{x}\in\mathbb{R}^p} \frac{1}{2}\|\mathbf{x}-\mathbf{u}\|^2 + \sum_{i=1}^{g} w_i f_i \tag{28}$$

Then, the Dykstra-like proximal algorithm can be summarized in Algorithm 2.

Algorithm 2 Dykstra-like Proximal Splitting Method

Set $\mathbf{x}_0 = \mathbf{u}$, $\mathbf{q}_{1,0}, \ldots, \mathbf{q}_{g,0} = \mathbf{x}_0$, $n = 0$
repeat
  for $i = 1, \ldots, g$ **do**
    $\mathbf{p}_{i,n} = \text{prox}_{f_i}\mathbf{q}_{i,n}$
  end for
  $\mathbf{x}_{n+1} = \sum_{i=1}^{g} w_i\mathbf{p}_{i,n}$
  for $i = 1, \ldots, g$ **do**
    $\mathbf{q}_{i,n+1} = \mathbf{x}_{n+1} + \mathbf{q}_{i,n} - \mathbf{p}_{i,n}$
  end for
  $n = n + 1$
until Convergence

The last piece of puzzle in Algorithm 2 is to solve $\mathbf{p} = \text{prox}_{f_i}\mathbf{q}$, defined as:

$$\mathbf{p} = \arg\min_{\mathbf{x}\in\mathbb{R}^p} \frac{1}{2}\|\mathbf{x}-\mathbf{q}\|^2 + \lambda\|\mathbf{x}_{G_i}\|$$

Clearly, we have $\mathbf{p}_{\overline{G}_i} = \mathbf{q}_{\overline{G}_i}$. For index set G_i, a close form solution is known to exist:

$$\mathbf{p}_{G_i} = \frac{\max(\|\mathbf{q}_{G_i}\| - \lambda, 0)}{\|\mathbf{q}_{G_i}\|}\mathbf{q}_{G_i}.$$

Thus, at each iteration, we have a closed-form solution.

C. Alternating Direction Method of Multipliers for Computing the Proximal Operator in (5)

Besides splitting the proximal operators, we can also bypass the difficulty brought by overlapping groups by introducing auxiliary variables, and reformulate (5) as:

$$\begin{aligned} \min_{\mathbf{x},\mathbf{z}} \quad & \frac{1}{2}\|\mathbf{x}-\mathbf{u}\|^2 + \lambda\sum_{i=1}^{g} w_i\|\mathbf{z}_i\| \\ \text{s.t.} \quad & \mathbf{z}_i = \mathbf{x}_{G_i}, \quad i = 1, \ldots, g \end{aligned} \tag{29}$$

We can therefore form the augmented Lagrangian as follows:

$$L_\rho(\mathbf{x}, \mathbf{z}, \mathbf{y}) = \frac{1}{2}\|\mathbf{x}-\mathbf{u}\|^2 + \lambda\sum_{i=1}^{g} w_i\|\mathbf{z}_i\| + \sum_{i=1}^{g}\mathbf{y}_i^T(\mathbf{z}_i - \mathbf{x}_{G_i}) + (\rho/2)\sum_{i=1}^{g}\|\mathbf{z}_i - \mathbf{x}_{G_i}\|^2.$$

The Alternating Direction Method of Multipliers (ADMM) consists of the following iterations:

$$\begin{aligned}\mathbf{x}^{k+1} &:= \arg\min_{\mathbf{x}} L_\rho(\mathbf{x}, \mathbf{z}^k, \mathbf{y}^k)\\ \mathbf{z}^{k+1} &:= \arg\min_{\mathbf{z}} L_\rho(\mathbf{x}^{k+1}, \mathbf{z}, \mathbf{y}^k)\\ \mathbf{y}_i^{k+1} &:= \mathbf{y}_i^k + \rho(\mathbf{z}_i^{k+1} - \mathbf{x}_{G_i}^{k+1})\end{aligned} \tag{30}$$

One nice property of ADMM is, each iterative step admits a closed-form solution. We define $\otimes$ as the point-wise product, $\odot$ as the point-wise division, $\mathbf{e}$ the p-dimensional vector with all ones, and the indicator vector $\tilde{\mathbf{e}}_i$ such that $\tilde{\mathbf{e}}_i(j) = 1$ if $j \in G_i$ and 0 otherwise. We further define $\tilde{\mathbf{y}}_i, \tilde{\mathbf{z}}_i \in \mathbb{R}^p$ such that $\tilde{\mathbf{y}}_i(G_i) = \mathbf{y}_i, \tilde{\mathbf{y}}_i(G_i^C) = 0$ and $\tilde{\mathbf{z}}_i(G_i) = \mathbf{z}_i, \tilde{\mathbf{z}}_i(G_i^C) = 0$. For updating $\mathbf{x}$, we have:

$$\frac{\partial}{\partial \mathbf{x}} L_\rho(\mathbf{x}, \mathbf{z}^k, \mathbf{y}^k) = \mathbf{x} - \mathbf{u} - \sum_{i=1}^g \tilde{\mathbf{y}}_i^k + \rho\left(\sum_{i=1}^g \tilde{\mathbf{e}}_i\right) \otimes \mathbf{x} - \rho\left(\sum_{i=1}^g \tilde{\mathbf{z}}_i^k\right)$$

and therefore,

$$\mathbf{x}^{k+1} = \left(\mathbf{u} + \sum_{i=1}^g \tilde{\mathbf{y}}_i^k + \rho \sum_{i=1}^g \tilde{\mathbf{z}}_i^k\right) \odot \left(\mathbf{e} + \rho \sum_{i=1}^g \tilde{\mathbf{e}}_i\right).$$

For updating $\mathbf{z}_i$, we use the sub-differential method: $\mathbf{z}^*$ is the optimal solution if and only if 0 belongs to the sub-differential set $\partial L_\rho(\mathbf{x}^{k+1}, \mathbf{z}^*, \mathbf{y}^k)$. Decouple the problem with respect to groups, we have:

$$0 \in \mathbf{z}_i^{k+1} - \mathbf{x}_{G_i}^{k+1} + \frac{1}{\rho}\mathbf{y}_i^k + \frac{\lambda w_i}{\rho}\partial\|\mathbf{z}_i^{k+1}\|$$

where

$$\partial\|\mathbf{z}_i^{k+1}\| = \begin{cases} \frac{\mathbf{z}_i^{k+1}}{\|\mathbf{z}_i^{k+1}\|} & \|\mathbf{z}_i^{k+1}\| \neq 0 \\ \{\mathbf{t} | \mathbf{t} \in \mathbb{R}^{|G_i|}, \|\mathbf{t}\| \leq 1\} & \|\mathbf{z}_i^{k+1}\| = 0. \end{cases}$$

Thus, we have:

$$\mathbf{z}_i^{k+1} = \frac{\max\{\|\tilde{\mathbf{x}}_{G_i}^{k+1}\| - \tilde{\lambda}_i, 0\}}{\|\tilde{\mathbf{x}}_{G_i}^{k+1}\|} \tilde{\mathbf{x}}_{G_i}^{k+1}$$

where

$$\tilde{\mathbf{x}}_{G_i}^{k+1} = \mathbf{x}_{G_i}^{k+1} - \frac{1}{\rho}\mathbf{y}_i^k, \quad \tilde{\lambda}_i = \frac{\lambda w_i}{\rho}.$$

Optimality conditions and stopping criterion The KKT conditions for (29) are primal feasibility:

$$\mathbf{z}_i^* - \mathbf{x}_{G_i}^* = 0 \tag{31}$$

and the dual feasibility:

$$\begin{aligned} 0 &= \mathbf{x}^* - \mathbf{u} - \sum_{i=1}^g \tilde{\mathbf{y}}_i^* \\ 0 &\in \lambda w_i \partial\|\mathbf{z}_i^*\| + \mathbf{y}_i^* \end{aligned} \tag{32}$$

Since $\mathbf{z}^{k+1}$ minimizes $L_\rho(\mathbf{x}^{k+1}, \mathbf{z}, \mathbf{y}^k)$, we have

$$\begin{aligned} 0 &\in \mathbf{z}_i^{k+1} - \mathbf{x}_{G_i}^{k+1} + \frac{1}{\rho}\mathbf{y}_i^k + \frac{\lambda w_i}{\rho}\partial\|\mathbf{z}_i^{k+1}\| \\ &= \frac{1}{\rho}\mathbf{y}_i^{k+1} + \frac{\lambda w_i}{\rho}\partial\|\mathbf{z}_i^{k+1}\| \end{aligned}$$

Therefore, the second condition in the dual feasibility is always satisfied, and the optimization comes down to attaining the primal and the first dual feasibility.

Define $r_i = \mathbf{z}_i - \mathbf{x}_{G_i}$. We have $\mathbf{y}_i^{k+1} = \mathbf{y}_i^k + \rho r^{k+1}$. Since $\mathbf{x}^{k+1}$ minimizes $L_\rho(\mathbf{x}, \mathbf{z}^k, \mathbf{y}^k)$, we have

$$\begin{aligned} 0 &= \mathbf{x}^{k+1} - \mathbf{u} - \sum_{i=1}^{g} \tilde{\mathbf{y}}_i^k + \rho \left(\sum_{i=1}^{g} \tilde{\mathbf{e}}_i \right) \otimes \mathbf{x}^{k+1} - \rho \left(\sum_{i=1}^{g} \tilde{\mathbf{z}}_i^k \right) \\ &= \mathbf{x}^{k+1} - \mathbf{u} - \sum_{i=1}^{g} \tilde{\mathbf{y}}_i^{k+1} + \rho \left(\sum_{i=1}^{g} (\tilde{\mathbf{z}}_i^{k+1} - \tilde{\mathbf{z}}_i^k) \right) \end{aligned}$$

or equivalently,

$$\rho \left(\sum_{i=1}^{g} (\tilde{\mathbf{z}}_i^k - \tilde{\mathbf{z}}_i^{k+1}) \right) = \mathbf{x}^{k+1} - \mathbf{u} - \sum_{i=1}^{g} \tilde{\mathbf{y}}_i^{k+1}.$$

This means that the quantity

$$s^{k+1} = \rho \left(\sum_{i=1}^{g} (\tilde{\mathbf{z}}_i^{k+1} - \tilde{\mathbf{z}}_i^k) \right)$$

can be viewed as the residual for the first dual feasibility. Paired with the primal residual r^{k+1}, we can terminate the algorithm by checking whether they are small enough.

D. Alternating Direction Method of Multipliers for Solving Overlapping Group Lasso

Using the least squared loss and observing that ℓ_1 norm is a special case of (2), we can rewrite the overlapping group lasso problem (1) as:

$$\begin{aligned} \min_{\mathbf{x},\mathbf{z}} \quad & \frac{1}{2}\|A\mathbf{x} - \mathbf{u}\|^2 + \lambda \sum_{i=1}^{g} w_i \|\mathbf{z}_i\| \\ \text{s.t.} \quad & \mathbf{z}_i = \mathbf{x}_{G_i} \end{aligned}$$

We can therefore form the augmented Lagrangian as follows:

$$L_\rho(A\mathbf{x}, \mathbf{z}, \mathbf{y}) = \frac{1}{2}\|A\mathbf{x} - \mathbf{u}\|^2 + \lambda \sum_{i=1}^{g} w_i \|\mathbf{z}_i\| + \sum_{i=1}^{g} \mathbf{y}_i^T (\mathbf{z}_i - \mathbf{x}_{G_i}) + (\rho/2) \sum_{i=1}^{g} \|\mathbf{z}_i - \mathbf{x}_{G_i}\|^2$$

The Alternating Direction Method of Multipliers (ADMM) consists of the iterations:

$$\begin{aligned} \mathbf{x}^{k+1} &:= \arg\min_{\mathbf{x}} L_\rho(\mathbf{x}, \mathbf{z}^k, \mathbf{y}^k) \\ \mathbf{z}^{k+1} &:= \arg\min_{\mathbf{z}} L_\rho(\mathbf{x}^{k+1}, \mathbf{z}, \mathbf{y}^k) \\ \mathbf{y}_i^{k+1} &:= \mathbf{y}_i^k + \rho(\mathbf{z}_i^{k+1} - \mathbf{x}_{G_i}^{k+1}) \end{aligned}$$

We define $\mathbf{e}$ the p-dimensional vector with all ones, and the indicator vector $\tilde{\mathbf{e}}_i$ such that $\tilde{\mathbf{e}}_i(j) = 1$ if $j \in G_i$ and 0 otherwise. We further define $\tilde{\mathbf{y}}_i, \tilde{\mathbf{z}}_i \in \mathbb{R}^p$ such that $\tilde{\mathbf{y}}_i(G_i) = \mathbf{y}_i, \tilde{\mathbf{y}}_i(G_i^C) = 0$ and $\tilde{\mathbf{z}}_i(G_i) = \mathbf{z}_i, \tilde{\mathbf{z}}_i(G_i^C) = 0$. For updating $\mathbf{x}$, we have:

$$\frac{\partial}{\partial \mathbf{x}} L_\rho(\mathbf{x}, \mathbf{z}^k, \mathbf{y}^k) = A^T A\mathbf{x} - A^T\mathbf{u} - \sum_{i=1}^{g} \tilde{\mathbf{y}}_i^k + \rho \left(\sum_{i=1}^{g} \tilde{\mathbf{e}}_i \right) \otimes \mathbf{x} - \rho \left(\sum_{i=1}^{g} \tilde{\mathbf{z}}_i^k \right)$$

and therefore, the update for $\mathbf{x}^{k+1}$ involves solving the following linear system:

$$\tilde{A}\mathbf{x} = \tilde{b},$$

where

$$\begin{aligned} \tilde{A} &= A^T A + \text{diag} \left(\rho \sum_{i=1}^{g} \tilde{\mathbf{e}}_i \right) \\ \tilde{b} &= A^T\mathbf{u} + \sum_{i=1}^{g} \tilde{\mathbf{y}}_i^k + \rho \sum_{i=1}^{g} \tilde{\mathbf{z}}_i^k \end{aligned}$$

Please note that, for a given problem, $\tilde{A}$ is fixed. Therefore, for moderate size problems, we can save the Cholesky decomposition of $\tilde{A}$ such that the linear system can be solved very fast in each iteration. For large (high dimensional) problems, the storage of $\tilde{A}$ might not be practical. However, since we can calculate $\tilde{A}\mathbf{x}$ without having to calculate $\tilde{A}$, methods such as Preconditioned Conjugate Gradient (PCG) or BB method can be applied.

For updating $\mathbf{z}_i$, we use the sub-differential method: $\mathbf{z}^*$ is the optimal solution if and only if 0 belongs to the sub-differential set $\partial L_\rho(\mathbf{x}^{k+1}, \mathbf{z}^*, \mathbf{y}^k)$. Decouple the problem with respect to groups, we have:

$$0 \in \mathbf{z}_i^{k+1} - \mathbf{x}_{G_i}^{k+1} + \frac{1}{\rho}\mathbf{y}_i^k + \frac{\lambda w_i}{\rho}\partial\|\mathbf{z}_i^{k+1}\|$$

where

$$\partial\|\mathbf{z}_i^{k+1}\| = \begin{cases} \frac{\mathbf{z}_i^{k+1}}{\|\mathbf{z}_i^{k+1}\|} & \|\mathbf{z}_i^{k+1}\| \neq 0 \\ \{\mathbf{t} | \mathbf{t} \in \mathbb{R}^{|G_i|}, \|\mathbf{t}\| \leq 1\} & \|\mathbf{z}_i^{k+1}\| = 0. \end{cases}$$

Thus, we have:

$$\mathbf{z}_i^{k+1} = \frac{\max\{\|\tilde{\mathbf{x}}_{G_i}^{k+1}\| - \tilde{\lambda}_i, 0\}}{\|\tilde{\mathbf{x}}_{G_i}^{k+1}\|}\tilde{\mathbf{x}}_{G_i}^{k+1}$$

where

$$\tilde{\mathbf{x}}_{G_i}^{k+1} = \mathbf{x}_{G_i}^{k+1} - \frac{1}{\rho}\mathbf{y}_i^k, \quad \tilde{\lambda}_i = \frac{\lambda w_i}{\rho}.$$

E. Additional Experiments

To illustrate the scalability of our proposed method, we also evaluate our method using numbers (p) of genes larger than 2000. The results are summarized in Table 2.

Table 2: Scalability study of the proposed FoGLasso algorithm under different numbers (p) of genes involved. The reported results are the total computational time (seconds) including all nine regularization parameter values.

p	3000	4000	5000	6000	7000	8141
pathways	37.6	48.3	62.5	68.7	86.2	99.7
edges	58.8	84.8	102.7	140.8	173.3	247.8