Faster Training Algorithms for Structured Sparsity-Inducing Norm

Bin Gu[*†], **Xingwang Ju**[*], **Xiang Li**[‡] and **Guansheng Zheng**[*]
[*]School of Computer & Software, Nanjing University of Information Science & Technology, P.R.China
[†]Department of Electrical & Computer Engineering, University of Pittsburgh, USA
[‡]Computer Science Department, University of Western Ontario, Canada
big10@pitt.edu, XwangJu@163.com, lxiang2@uwo.ca, zgs@nuist.edu.cn

Abstract

Structured-sparsity regularization is popular for sparse learning because of its flexibility of encoding the feature structures. This paper considers a generalized version of structured-sparsity regularization (especially for l_1/l_∞ norm) with arbitrary group overlap. Due to the group overlap, it is time-consuming to solve the associated proximal operator. Although Mairal *et al.* have proposed a network-flow algorithm to solve the proximal operator, it is still time-consuming, especially in the high-dimensional setting. To address this challenge, in this paper, we have developed a more efficient solution for l_1/l_∞ group lasso with arbitrary group overlap using inexact proximal gradient method. In each iteration, our algorithm only requires to calculate an inexact solution to the proximal sub-problem, which can be done efficiently. On the theoretic side, the proposed algorithm enjoys the same global convergence rate as the exact proximal methods. Experiments demonstrate that our algorithm is much more efficient than the network-flow algorithm, while retaining the similar generalization performance.

1 Introduction

In machine learning, sparse linear models (SLM) [Neumaier and Groeneveld, 1998] emphasizes the principle that a simpler model representation should be preferred over more complicated ones [Bach *et al.*, 2012]. As the most basic SLM, Lasso [Friedman *et al.*, 2008] achieves sparsity by penalizing the l_1-norm of the weight vector. Lasso has been proven to be effective in various high-dimensional learning tasks through a number of technical means (such as [Kohavi and others, 1995; Gu and Ling, 2015; Gu *et al.*, 2017b]). However, since the modulus of each feature is penalized equally, Lasso cannot encode feature correlations such as groups, graphs or other patterns that one may induce from prior knowledge.

To address this challenge, various norms that encourages a certain structured-sparsity have emerged. For example, group lasso [Jacob *et al.*, 2009], where features are partitioned into predefined groups according to prior knowledge. l_1-norm penalties [Efron *et al.*, 2004; Nesterov and others, 2007] are given to each group of features instead. While providing enhanced model flexibility, a group lasso penalty often causes non-smooth programs. As a result, proximal gradient methods [Beck and Teboulle, 2009] are the most commonly used technique for its optimization. In particular, each pivot step of a proximal gradient algorithm requires to solve the following proximal subproblem:

$$\text{Prox}_{\gamma h}(y) = \arg\min_{x\in\mathbb{R}^n} \frac{1}{2\gamma}\|x-y\|^2 + h(x) \qquad (1)$$

where γ is the stepsize set by a line-search procedure [Beck and Teboulle, 2009] or manually, and $h(x)$ is a non-smooth and convex regularization function. (1) will be called in every iteration of a gradient-based model update. Consequently, the success of a proximal gradient algorithm as a whole relies on whether the proximal operator (1) can be solved efficiently. However, this requirement may not always be easily satisfied. Many research efforts have been devoted to devising fast proximal operator solvers for different types of norms that induce structured-sparsity. For example, [Yuan *et al.*, 2011; Villa *et al.*, 2014; Hegde *et al.*, 2015; Kim and Xing, 2010; Yu *et al.*, 2012] to name a few. However, these solvers are often restricted to a specific formulation, and are not well-generalized. Moreover, many of them are effective only when learning a moderate number of feature dimensions and groups, and are hardly scalable to a higher-dimensional scenario.

In this paper, we propose a more general solver to the proximal subproblem (1). We demonstrate its effectiveness by studying the l_1/l_∞ norm group lasso with arbitrary group overlap. The l_1/l_∞ norm group lasso is one of the most popular group lasso formulations. We have seen its applications in background subtraction [Mairal *et al.*, 2010], dictionary learning [Mairal *et al.*, 2010], image annotation [Quattoni *et al.*, 2009], transfer learning [Quattoni *et al.*, 2008], and so on. This norm takes the formulation $\sum_{g\in\phi} \eta_g \parallel x_g\parallel_\infty$, where g denotes a group. A more flexible formulation of l_1/l_∞ norm group lasso, is where arbitrary group overlap is allowed. For this scenario, [Mairal *et al.*, 2010] have proposed a network-flow algorithm to calculate its proximal operator. Together with a fast proximal gradient framework [Jenatton *et al.*, 2010], the effectiveness of this algorithm has been verified on datasets with $m \times n$ of features and $n-2$ groups. However, the aim of the network-flow algorithm is to calculate an exact

solution to the proximal operator, which may be slow when the number of features and feature groups are much larger.

To overcome this performance bottleneck, our proposed solution is to use an Inexact Proximal Gradient (IPG) algorithm [Schmidt *et al.*, 2011]. Basically, an IPG algorithm only calculates an inexact solution to the proximal problem in each gradient iteration, which can be done efficiently even with basic optimizers such as gradient descent. Inexact methods [Gu *et al.*, 2018c] and [Schmidt *et al.*, 2011] have proved that it is possible to use IPG on regularized risk minimization problem with non-smooth norm in certain assumptions. In theory, IPG will enjoy similar global convergence rate as the basic exact proximal gradient method. This means that IPG needs a similar number of outer iterations to achieve a solution with certain accuracy, even though the proximal operator is only calculated in an *inexact* fashion. Leveraging this theoretic result, we develop an IPG program for the l_1/l_∞ overlap group norm problem. With experiments, we show that the proposed algorithm can handle high-dimensional problems where the original network-flow algorithm becomes hardly scalable. We follow the work of [Gu *et al.*, 2018c], by using a sub-gradient algorithm to solve the proximal sub-problem approximately, i.e., when computing the proximal operator with a tolerated error.

On the theoretic side, the proposed algorithm enjoys the same global convergence rate as the exact proximal methods. Our method achieves higher time efficiency. To the best of our knowledge, the network-flow algorithm can deal with l_1/l_∞ group lasso with a relatively simple group overlap structure. However, for high-dimensional and complicated overlap groups, our method is much more efficient, while retaining the similar generalization performance as the network-flow algorithm.

2 Structured Sparsity-Inducing Norm

In this paper, we consider l_1/l_∞-norm as the structured sparsity-inducing norm, and give a brief review of the ProxFlow algorithm [Mairal *et al.*, 2010] which can solve the l_1/l_∞-norm.

2.1 l_1/l_∞-norm

As shown in [Mairal *et al.*, 2010], the structured sparsity-inducing norm (also called as l_1/l_∞-norm) takes the form:

$$h(x)=\lambda\sum_{g\in\phi}\eta_g\max_{i\in g}|x_t| =\lambda\sum_{g\in\phi}\eta_g \parallel x_g\parallel_\infty \qquad (2)$$

where ϕ is a set of groups, g denotes a group in ϕ, x_t denotes the i-th dimension of the vector $x\in\mathbb{R}^n$, the vector $x_g\in\mathbb{R}^{|g|}$ denotes the features of x make up of group g, and the η_{g} denotes the positive weight for each group. l_1/l_∞-norm [Mairal *et al.*, 2010] is one of the most important joint regularization technique with many applications [Quattoni *et al.*, 2009; Gu *et al.*, 2017a] which imply that the l_1/l_∞-norm becomes the current research hotspot. In this paper, we consider a general case where arbitrary group overlap is allowed as discussed before.

2.2 ProxFlow Algorithm

[Mairal *et al.*, 2010] proposed the ProxFlow algorithm to exactly compute the proximal operator with l_1/l_∞-norm penalty by converting this problem to a quadratic min-cost flow problem. In the computation of ProxFlow algorithm, it reformulates the dual problem of the proximal projection step as a quadratic min-cost flow algorithm based on a canonical graph model $G(V,E)$ [Mairal *et al.*, 2010]. Note that V is the set of vertexes combined by features and groups ($|V| = |\phi| + n$), E represents the relationships of vertexes($|E| = |\phi| + \sum_{g\in\phi}|g| + n$). Thus, we can calculate the projection step effectively and it's worst-case complexity is $\mathrm{O}(|V|^2|E|^{\frac{1}{2}})$.

However, in reality, we sometimes encounter the problems with high dimensional data and overlapped groups. For instance, the OSCAR structured sparsity-inducing norm ($h(x) = \lambda_1\|x\|_1 + \lambda_2\sum_{i<j}\max\{|x_t|,|x_j|\}$) [Zhong and Kwok, 2012] is a special case of Eq. (2). OSCAR setup has $O(n^2)$ number of groups, and the canonical graph in the ProxFlow algorithm will has $|V| = |E| = O(n^2)$. Thus, the time complexity for each projection step is $O(n^5)$ which is unacceptable in practical applications. In order to deal with high-dimensional problems and complicated overlap groups efficiently, we follow [Gu *et al.*, 2018c; Schmidt *et al.*, 2011] to develop the inexact proximal gradient algorithms to solve the l_1/l_∞-norm.

3 Inexact Proximal Gradient Methods

Since [Schmidt *et al.*, 2011] proposed inexact proximal gradient algorithms, inexact proximal gradient algorithms have been widely used in various applications. In this paper, we will apply the inexact proximal gradient algorithms [Schmidt *et al.*, 2011] to solve the structured sparsity-inducing norm as introduced above.

Before presenting our inexact proximal gradient algorithms, we first formulate the optimization problem with the structured sparsity-inducing norm (2) considered in this paper as follows.

$$\min_{x\in\mathbb{R}^n} f(x)+h(x)=\sum_{i=1}^{l} f_i(x)+\lambda\sum_{g\in\phi}\eta_g \parallel x_g\parallel_\infty \qquad (3)$$

where f_i is the loss function on the i-th sample.

Then, we define the proximal problem associated to the structured sparsity-inducing norm (2) as follows.

$$Q(x;y)=\frac{1}{2\gamma}\|x-y\|^2+\lambda\sum_{g\in\phi}\eta_g \parallel x_g\parallel_\infty \qquad (4)$$

Based on the proximal problem (4), we define the inexact proximal operator $\mathrm{Prox}^{\varepsilon}_{\gamma\mathrm{h}}(y)$ associated to the structured sparsity-inducing norm (2) as follows.

$$\begin{aligned} & x\in \mathrm{Prox}^{\varepsilon}_{\gamma\mathrm{h}}(y) \qquad (5)\\ = \;& \left\{x\in R^n : Q(x;y)\le \varepsilon+\min_z Q(z;y)\right\}\end{aligned}$$

where ε denotes the error of proximal operator in the calculation. The error of the proximal operator can be controlled

based on the dual gap which will be discussed in Section 3.1. Based on the inexact proximal operator (5), we introduce our basic inexact proximal gradient algorithm (IPG, see Section 3.2) and accelerated inexact proximal gradient algorithm (AIPG, see Section 3.3).

3.1 Inexact Subgradient Algorithm for Solving (5)

It is hard to find a closed-form solution to the proximal problem (4) with the l_1/l_∞-norm penalty. As mentioned before, ProxFlow algorithm [Mairal *et al.*, 2010] provides a recursive algorithm to exactly compute the proximal problem (4). However, as discussed in Section 2.2, it is time-consuming to give the solution for the l_1/l_∞-norm with high dimensions and complicated overlapping groups. In this paper, instead of finding an exact solution to the proximal problem (4), we will use a subgradient descent procedure [Gu *et al.*, 2018c] to find a solution to the inexact proximal operator $\text{Prox}^{\varepsilon}_{\gamma \text{h}}(y)$. We consider using the duality gap to check whether the solution of the subgradient gradient algorithm satisfies the predefined error ε. Specifically, the duality gap is formulated as

$$G(x_t; y) = Q(x_t; y) - \tilde{Q}(\alpha_t; y) \tag{6}$$

where α is the dual variable, and $\tilde{Q}(\alpha_t; y)$ is the dual of $Q(x_t; y)$. There are three pivotal steps for the subgradient descent algorithm.

1. Compute a subgradient $f_t^{'} \in \partial Q(x_t; y)$.
2. Update the solution $x_{t+1} \leftarrow x_t - \gamma' f_t^{'}$.
3. Check the duality gap $G(x_t; y)$.

We summarize the inexact subgradient descent algorithm in Algorithm 1. In the following, we mainly discuss the first and third steps.

Compute the Subgradient: When each group g in ϕ is a single feature, the l_∞ falls back the l_1-norm. The problem became simple, and the subgradient of each feature is $\partial h(x_i) = \eta_g[-1, 1]$, if $x_i = 0$. Otherwise, $\partial h(x_i) = \eta_g \frac{x_i}{|x_i|}$. But for each group with more than one features in $\{1, \ldots, n\}$, the subgradient of equation (2) can be formulated as follows.

$$\partial h(x_t) = \sum_{g \in \phi} \eta_{\text{g}} \partial \|x_g^{\max}\|_1 \tag{7}$$

where $x_g^{\max}$ denotes the coordinate of x_g with the largest absolute value. Thus, the subgradient $\partial Q(x; y)$ can be formulated as:

$$\partial Q(x; y) = \frac{1}{\gamma}(x - y) + \sum_{g \in \phi} \eta_{\text{g}} \partial \|x_g^{\max}\|_1 \tag{8}$$

Compute the duality gap: The proximal problem $Q(x; y)$ is a convex problem. Thus, we can guarantee the solution x is a ε-approximation solution with $Q(x_t; y) - Q(x^*; y) \leq \varepsilon$ by the duality gap $G(x_t; y) = Q(x_t; y) - \tilde{Q}(\alpha_t; y) \leq \varepsilon$, where x^* is an optimal solution of $Q(x; y)$. This conclusion holds because $Q(x^*; y) - Q(x_t; y) \leq G(x_t; y)$ [Boyd and Vandenberghe, 2004].

As mentioned in [Mairal *et al.*, 2010], the dual function $\tilde{Q}(\alpha; y)$ can be formulated as

$$\tilde{Q}(\alpha; y) = \sup_z z^T \alpha - \frac{1}{2\gamma} \|x - y\|^2, \quad s.t.\ \Omega^*(\alpha) \leq \lambda \tag{9}$$

where $\Omega^*(\alpha)$ is the dual norm of l_1/l_∞-norm $\sum_{g \in \phi} \eta_g \| x_g\|_\infty$. Given a primal solution x, we have that $\alpha = y - x$. Therefore, evaluating the duality gap requires to compute efficiently Ω^* in order to find a feasible dual variable ξ. This is equivalent to solving another network flow problem, based on the following variational formulation:

$$\begin{aligned} \Omega^*(\alpha) = \min_{\xi \in \mathbb{R}^{n \times |\phi|}} \quad & \tau \\ s.t. \quad & \sum_{g \in \phi} \xi^g = \alpha, \quad \forall g \in \phi, \\ & \|\xi^g\|_1 \leq \lambda \eta_g \ \ with \ \ \xi_j^g = 0, if \ \ j \notin g. \end{aligned} \tag{10}$$

[Mairal *et al.*, 2010] provides an algorithm to compute the dual norm.

Algorithm 1 Inexact Subgradient Descent Algorithm

Input: Error ε, stepsize γ', x_{i-1}, loop number k.
Output: x_s.

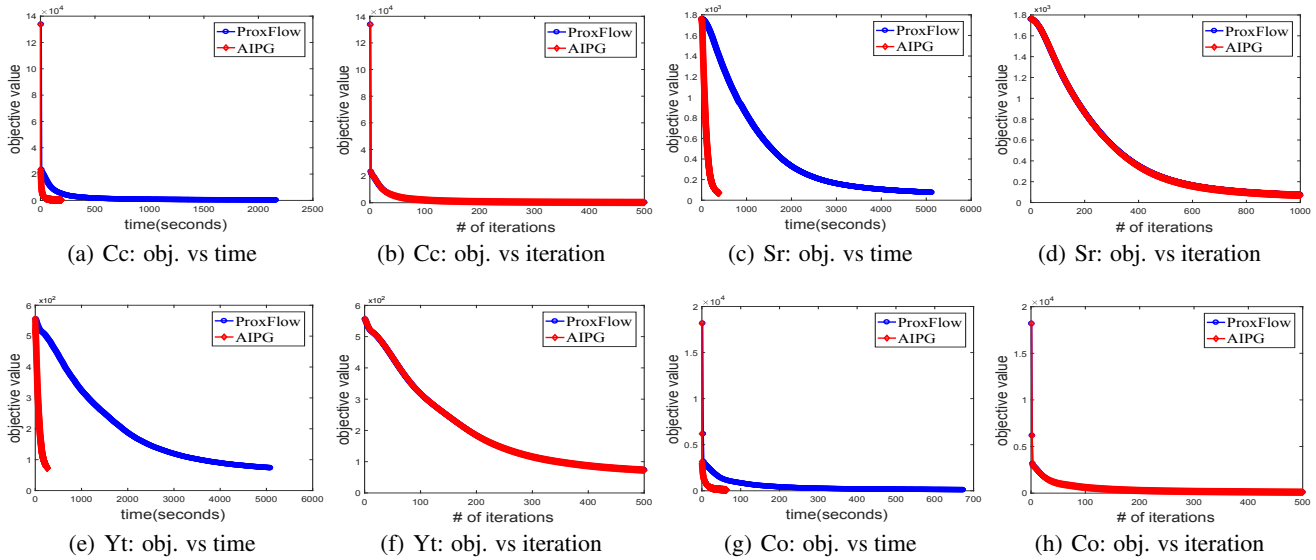
```
Initialize x_0 = x_{i-1}, k = 0, Flag = 1.
for Flag do
    Compute a subgradient f_t' ∈ ∂Q(x_t; y).
    Update x_{t+1} ← x_t − γ' f_t'.
    t ← t + 1.
    if t%k = 0 then
        Flag=G(x_t; y) > ε.
    end if
end for
```

3.2 IPG Algorithm

[Beck and Teboulle, 2009] proposed the exact proximal gradient algorithm for solving linear inverse problems. The inexact proximal gradient algorithm (i.e., IPG) was first proposed by [Bach *et al.*, 2012]. The authors have proved that when the $\{\varepsilon_t\}_{t=1}^{1,\ldots,s}$ is a decreasing sequence and satisfies $\sum_{i=1}^{s} \varepsilon_t < \infty$ (Theorem 1), the convergence rate of IPG can has $O(\frac{1}{T})$ ([Schmidt *et al.*, 2011; Gu *et al.*, 2018c]) which is the same as the exact basic method. Here we apply the basic inexact proximal gradient algorithm (i.e., IPG) to solve the problem (3) with the structured sparsity-inducing norm. Specifically, we give our IPG algorithm in Algorithm 2.

3.3 AIPG Algorithm

Nesterov's accelerated proximal gradient method has faster convergence rate than the basic proximal gradient method. Unlike the exact accelerated proximal gradient (APG) method [Beck and Teboulle, 2009] with $O(\frac{1}{T^2})$ convergence rate, the convergence rate of inexact proximal gradient method is relevant to $\{\sqrt{\varepsilon_t}\}_{t=1}^{1,\ldots,s}$. AIPG achieves $O(\frac{1}{T^2})$ convergence rate only if $\{\sqrt{\varepsilon_t}\}_{t=1}^{1,\ldots,s}$ satisfy a certain summable condition [Schmidt *et al.*, 2011]. Furthermore, in [Gu *et al.*, 2018c], the authors analyzed the convergence rate for the *non-convex* case in different ranges. Here we apply the accelerated inexact proximal gradient algorithm to solve the problem (3) with the structured sparsity-inducing norm, and proposed our AIPG algorithm in Algorithm 3.



(a) Cc: obj. vs time (b) Cc: obj. vs iteration (c) Sr: obj. vs time (d) Sr: obj. vs iteration

(e) Yt: obj. vs time (f) Yt: obj. vs iteration (g) Co: obj. vs time (h) Co: obj. vs iteration

Figure 1: Comparison of convergence speed for real-world datasets for regression in Network setup.

Algorithm 2 IPG Algorithm

Input: Iteration number s, error ε_t $(t = 1, \ldots, s)$, stepsize $\gamma < \frac{1}{L}$.
Output: x_s.
1: Initialize x_0.
2: **for** $t = 1, 2, \ldots, s$ **do**
3: $x_t \in \text{Prox}_{\gamma \text{h}}^{\varepsilon_t}(x_{t-1} - \gamma \nabla f(x_{t-1}))$.
4: **end for**

Algorithm 3 AIPG Algorithm

Input: Iteration numbers, error ε_t $(t = 1, \ldots, s)$, stepsize $\gamma < \frac{1}{L}$, $k_1 = 2$, $k_0 = 1$.
Output: x_s.
1: Initialize x_0, and set $x_1 = z_1 = x_0$.
2: **for** $t = 1, 2, \ldots, s$ **do**
3: $y_t = x_t + \frac{k_{t-1}-1}{t_t}(k_t - k_{t-1})$.
4: $x_{i+1} \in \text{Prox}_{\gamma \text{h}}^{\varepsilon_t}(y_t - \gamma \nabla f(y_t))$.
5: $k_{t+1} = \frac{\sqrt{4{k_t}^2+1}+1}{2}$.
6: **end for**

3.4 Convergence Analysis

As mentioned above, the convergence rate of inexact proximal gradient methods are proved by [Gu *et al.*, 2018c; Schmidt *et al.*, 2011]. With the L-Lischitz smooth, ε-approximate subdifferential and the ε-KL property (Definition 1, 2 in [Gu *et al.*, 2018c]), IPG and AIPG will eventually converge to a stationary point if $\{\varepsilon_i\}_{i=1}^{1,\ldots,s}$ is a decreasing sequence and $\sum_{i=1}^{s} \varepsilon_t < \infty$. To make the paper self-contained, we present this conclusion in Theorem 1.

Theorem 1 *For IPG and AIPG algorithms, if* $\{\varepsilon_t\}_{t=1}^{1,\ldots,s}$ *is a decreasing sequence and* $\sum_{t=1}^{s} \varepsilon_t < \infty$*, we have that* $\mathbf{0} \in lim_{t\to\infty} \nabla f(x_t) + \partial_{\varepsilon_t} h(x_t)$.

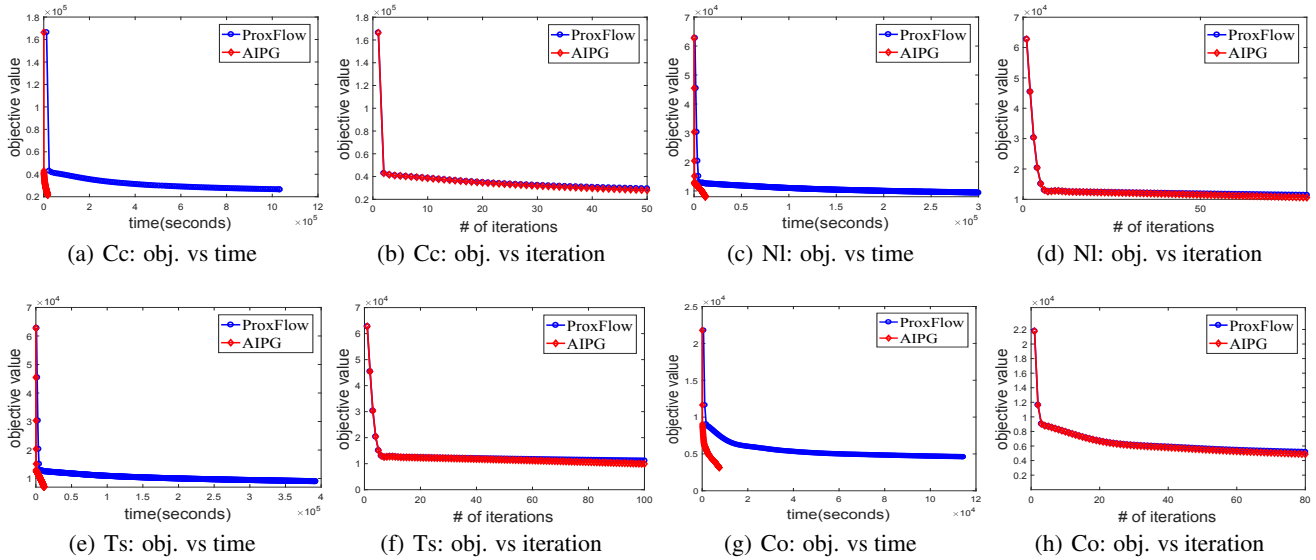
Convergence rate of IPG: As mentioned in Section 4 in [Schmidt *et al.*, 2011] and Theorem 2 in [Gu *et al.*, 2018c], the convergence rate of IPG has $O(\frac{1}{T})$ for convex and non-convex settings when the $\{\sqrt{\varepsilon_i}\}_{i=1}^{1,\ldots,s}$ is summable.
Convergence rate of AIPG: Unlike the convergence of basic proximal method, the convergence rate of accelerated method is complicated for either the convex or non-convex case. [Schmidt *et al.*, 2011] proved that AIPG can have the $O(\frac{1}{T^2})$ convergence rate if $\{\sqrt{\varepsilon_i}\}_{i=1}^{1,\ldots,s}$ be summable for the convex setting. For non-convex setting, [Gu *et al.*, 2018c] proved that AIPG converges in a finite number of iterations when $\theta = 1$, in a linear rate when $\theta \in [\frac{1}{2}, 1)$ and at least a sublinear rate when $\theta \in (0, \frac{1}{2})$, where θ is a parameter related to the desingularising function.

4 Experiments

In order to test the performance of our algorithms, we implement our inexact proximal gradient algorithms in MATLAB. To compare the run-time in the same platform, we also implement the ProxFlow algorithm [Mairal *et al.*, 2010] in Matlab. In the following, we first give the experimental setup, then present our experimental results and analysis.

4.1 Experimental Setup

Design of Experiments: We apply the following three group setups to validate the effectiveness of our inexact algorithm. In experiments, the outer loop iteration is selected from $\{300, 500, 1000\}$ to guarantee convergence. The value of stepsize γ is selected from $\{10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}\}$ to satisfy $\gamma \leq \frac{1}{L}$. The λ is set 0.1. The initial vector x_i has 20% percent nonzero components, randomly selected, and uniformly generated between $[-1, 1]$ for normalization. The weight η_g for each group is also randomly generated between $[0, 1]$.



(a) Cc: obj. vs time (b) Cc: obj. vs iteration (c) Nl: obj. vs time (d) Nl: obj. vs iteration

(e) Ts: obj. vs time (f) Ts: obj. vs iteration (g) Co: obj. vs time (h) Co: obj. vs iteration

Figure 2: Comparison of convergence speed for real-world datasets in OSCAR setup.

1. **Network setup:** To compare with ProxFlow method, we use the experimental setup of [Mairal *et al.*, 2010]. As mentioned in [Mairal *et al.*, 2010], for overlap groups ϕ, every 3 contiguous features combined into one group, then we have $n-2$ groups.
2. **OSCAR setup:** The experimental setup of [Mairal *et al.*, 2010] is a bit simple. As mentioned in [Gu *et al.*, 2018c], the OSCAR problem can be solved by inexact algorithm efficiently than the exact proximal gradient method [Zhong and Kwok, 2012], hence, we also use the OSCAR setup with $n^2/2$ overlap groups.
3. **Random setup:** The OSCAR setup is too complicated that with a large number of groups and the Network setup is a bit simple that the features of each group are successive, therefore we also implement the Random setup that with the $2n$ groups and the number of features for each group is randomly selected between 5 and 20. The features for each group are randomly selected.

Datasets: The experiment were done on eight real-world datasets as shown in Table 1. Specifically, the Cardiac dataset was collected from 3360 MRI images by hospital and each image has 400 pixels. The dataset is to predict the area of left ventricle. The Nartual, Tesdata, Yearst, Sector and Realsim datasets are from `http://www.mldata.org/repository/data/`. The Coil20 dataset is from `http://www.cs.columbia.edu/CAVE/software/`. The Movielen100k dataset is from `http://archive.ics.uci.edu/ml/datasets.html`. Note that, the alphabets in the $(\cdot)$ are the abbreviation of the name of the corresponding dataset.

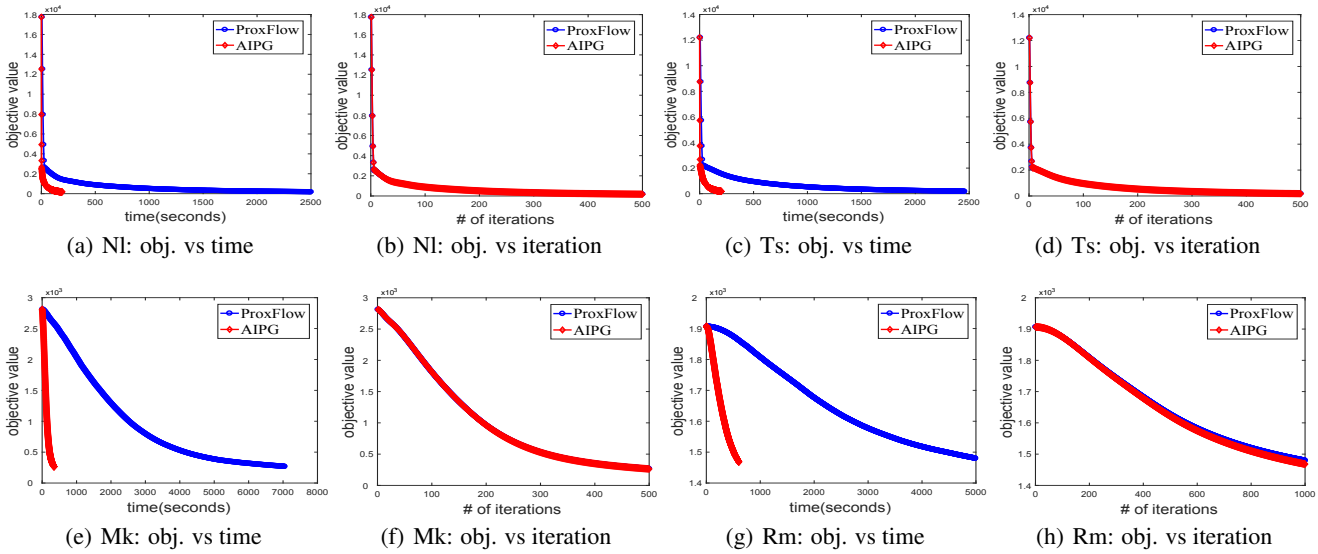
4.2 Results

Network setup: In figure 1, we compare our method with the Proxflow method. The figures 1a, 1c, 1e, 1g present the convergence rates of objective value with respect to the running time. Figures 1b, 1d, 1f, 1h present the convergence value with respect to iteration in the Network setup. From these presented experimental results of these real-world datasets, our methods can provide more flexible algorithms for l_1/l_∞-norm than ProxFlow algorithm. More significantly, our algorithm is **faster** than the exact method, especially for those high-dimensional datasets. With the increase of the dimension, the time cost of our algorithm only takes less time compared with ProxFlow. This is because our subgradient algorithm is more efficient for solving the proximal problem compared to an exact solver. Meanwhile, as for the number of outer iterations, our algorithm maintains the same convergence rate as the exact methods.

Datasets	Sample size(m)	Features(n)
Cardiac(Cc)	3360	1600
Sector(Sr)	9619	1000
Yearst(Yt)	103	2417
Nartual(Nl)	1024	1000
Tesdata(Ts)	1024	1000
Coil20(Co)	2580	700
Movielen100k(Mk)	943	1682
Realsim(Rm)	20958	1000

Table 1: The real-world datasets used in experiments.

OSCAR setup: Figures 2a, 2c, 2e and 2g present the convergence rates of objective value with respect to the running time. Figures 2b, 2d, 2f and 2h present the convergence value with respect to the iteration of our inexact algorithm and ProxFlow for regression problem in the OSCAR setup. We show that our algorithm can better deal with a large number of overlapping groups for the complexity of OSCAR setup and the convergence speed is **faster** than the ProxFlow algorithm. Hence, our algorithm has more advantages and can be more conducive to practical application.



(a) Nl: obj. vs time (b) Nl: obj. vs iteration (c) Ts: obj. vs time (d) Ts: obj. vs iteration

(e) Mk: obj. vs time (f) Mk: obj. vs iteration (g) Rm: obj. vs time (h) Rm: obj. vs iteration

Figure 3: Comparison of convergence speed for real-world datasets in Random setup.

Random setup: In figure 3, we compare our method with the Proxflow method. The figures 3a, 3c, 3e, 3g present the convergence rates of objective value with respect to the running time. Figures 3b, 3d, 3f, 3h present the convergence value with respect to iteration of our inexact algorithm and ProxFlow for regression problem in the Random setup. Our inexact method is **significantly faster** than the ProxFlow algorithm. From the figures, we can clearly confirm that our algorithm has a great advantage than the ProxFlow.

Summary of regression results: Based on the results, we verify that our inexact proximal gradient algorithm is more efficient for l_1/l_∞ penalty than ProxFlow algorithm. The results demonstrate our methods is more flexible for the structured sparsity-inducing norm, especially for complicated group structures. We also show the accuracy of our algorithm and the ProxFlow algorithm in figure 4. Our inexact proximal gradient methods perform better or at least equally compared to the exact methods. The experimental results verify the generalization performance of inexact proximal gradient methods on these datasets.

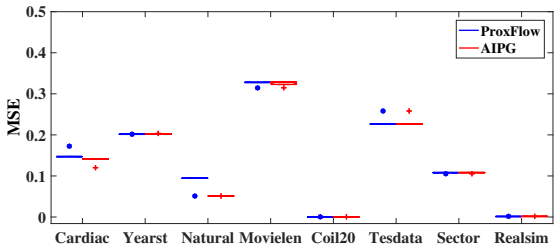


Figure 4: Testing MSEs of exact and inexact algorithms over 10 trials.

5 Conclusion

In this paper, we first intrduce the l_1/l_∞-norm penalty, then we propose using inexact proximal gradient methods to solve the l_1/l_∞-norm group lasso with arbitrary overlap group problem. We provide a sub-gradient algorithm to solve the proximal sub-problem approximately. Finally, numerical experimental results demonstrate that our subgradient algorithm to compute the proximal operator enjoys great efficiency with less time-cost and has the same convergence rate as the exact proximal method (ProxFlow, [Mairal *et al.*, 2010]). Especially for the high-dimensional case with complicated groups, our algorithm has an advantage over the ProxFlow algorithm and has a great effectiveness in the accuracy, which can better apply in real-world applications for overlapping group lasso problem. In future, we are interested in extending our algorithms to the asynchronous computation for inexact gradient updating algorithm [Gu *et al.*, 2018a; 2018b] and compositional problems [Huo *et al.*, 2018].

Acknowledgments

Dr. Heng Huang made scientific contributions to this paper. Unfortunately the conference chair doesn't allow us to change the author list. We acknowledge Dr. Heng Huang's contributions. This work was partially supported by the Natural Science Foundation of Jiangsu Province of China (No. BK20161534), Six talent peaks project (No. XYDXX-042) and the 333 Project (No. BRA2017455) in Jiangsu Province and the National Natural Science Foundation of China (No 61573191). BG and HH were partially supported by U.S. NSF-IIS 1302675, NSF-IIS 1344152, NSF-DBI 1356628, NSF-IIS 1619308, NSF-IIS 1633753, NIH R01 AG049371.

References

[Bach *et al.*, 2012] Francis Bach, Rodolphe Jenatton, Julien Mairal, Guillaume Obozinski, et al. Optimization with

sparsity-inducing penalties. *Foundations and Trends in Machine Learning*, 4(1):1–106, 2012.

[Beck and Teboulle, 2009] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

[Boyd and Vandenberghe, 2004] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[Efron *et al.*, 2004] Bradley Efron, Trevor Hastie, Iain Johnstone, Robert Tibshirani, et al. Least angle regression. *The Annals of statistics*, 32(2):407–499, 2004.

[Friedman *et al.*, 2008] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

[Gu and Ling, 2015] Bin Gu and Charles Ling. A new generalized error path algorithm for model selection. In *International Conference on Machine Learning*, pages 2549–2558, 2015.

[Gu *et al.*, 2017a] Bin Gu, Guodong Liu, and Heng Huang. Groups-keeping solution path algorithm for sparse regression with automatic feature grouping. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 185–193. ACM, 2017.

[Gu *et al.*, 2017b] Bin Gu, Victor S Sheng, Keng Yeow Tay, Walter Romano, and Shuo Li. Cross validation through two-dimensional solution surface for cost-sensitive svm. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1103–1121, 2017.

[Gu *et al.*, 2018a] Bin Gu, Zhouyuan Huo, and Heng Huang. Asynchronous doubly stochastic group regularized learning. In *International Conference on Artificial Intelligence and Statistics*, pages 1791–1800, 2018.

[Gu *et al.*, 2018b] Bin Gu, Xin Miao, Zhouyuan Huo, and Heng Huang. Asynchronous doubly stochastic sparse kernel learning. In *AAAI*, 2018.

[Gu *et al.*, 2018c] Bin Gu, De Wang, Zhouyuan Huo, and Heng Huang. Inexact proximal gradient methods for non-convex and non-smooth optimization. In *AAAI*, 2018.

[Hegde *et al.*, 2015] Chinmay Hegde, Piotr Indyk, and Ludwig Schmidt. A nearly-linear time framework for graph-structured sparsity. In *International Conference on Machine Learning*, pages 928–937, 2015.

[Huo *et al.*, 2018] Zhouyuan Huo, Bin Gu, Ji Liu, and Heng Huang. Accelerated method for stochastic composition optimization with nonsmooth regularization. In *AAAI*, 2018.

[Jacob *et al.*, 2009] Laurent Jacob, Guillaume Obozinski, and Jean-Philippe Vert. Group lasso with overlap and graph lasso. In *Proceedings of the 26th annual international conference on machine learning*, pages 433–440. ACM, 2009.

[Jenatton *et al.*, 2010] Rodolphe Jenatton, Julien Mairal, Guillaume Obozinski, and Francis R Bach. Proximal methods for sparse hierarchical dictionary learning. In *ICML*, number 2010, pages 487–494. Citeseer, 2010.

[Kim and Xing, 2010] Seyoung Kim and Eric P Xing. Tree-guided group lasso for multi-task regression with structured sparsity. 2010.

[Kohavi and others, 1995] Ron Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, volume 14, pages 1137–1145. Montreal, Canada, 1995.

[Mairal *et al.*, 2010] Julien Mairal, Rodolphe Jenatton, Francis R Bach, and Guillaume R Obozinski. Network flow algorithms for structured sparsity. In *Advances in Neural Information Processing Systems*, pages 1558–1566, 2010.

[Nesterov and others, 2007] Yurii Nesterov et al. Gradient methods for minimizing composite objective function, 2007.

[Neumaier and Groeneveld, 1998] Arnold Neumaier and Eildert Groeneveld. Restricted maximum likelihood estimation of covariances in sparse linear models. *Genetics Selection Evolution*, 30(1):3, 1998.

[Quattoni *et al.*, 2008] Ariadna Quattoni, Michael Collins, and Trevor Darrell. Transfer learning for image classification with sparse prototype representations. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[Quattoni *et al.*, 2009] Ariadna Quattoni, Xavier Carreras, Michael Collins, and Trevor Darrell. An efficient projection for l 1, ∞ regularization. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 857–864. ACM, 2009.

[Schmidt *et al.*, 2011] Mark Schmidt, Nicolas L Roux, and Francis R Bach. Convergence rates of inexact proximal-gradient methods for convex optimization. In *Advances in neural information processing systems*, pages 1458–1466, 2011.

[Villa *et al.*, 2014] Silvia Villa, Lorenzo Rosasco, Sofia Mosci, and Alessandro Verri. Proximal methods for the latent group lasso penalty. *Computational Optimization and Applications*, 58(2):381–407, 2014.

[Yu *et al.*, 2012] Guoshen Yu, Guillermo Sapiro, and Stéphane Mallat. Solving inverse problems with piecewise linear estimators: From gaussian mixture models to structured sparsity. *IEEE Transactions on Image Processing*, 21(5):2481–2499, 2012.

[Yuan *et al.*, 2011] Lei Yuan, Jun Liu, and Jieping Ye. Efficient methods for overlapping group lasso. In *Advances in Neural Information Processing Systems*, pages 352–360, 2011.

[Zhong and Kwok, 2012] Leon Wenliang Zhong and James T Kwok. Efficient sparse modeling with automatic feature grouping. *IEEE transactions on neural networks and learning systems*, 23(9):1436–1447, 2012.