An augmented ADMM algorithm with application to the generalized lasso problem

Yunzhang Zhu [*]
Department of Statistics, The Ohio State University

October 28, 2015

Abstract

In this article, we present a fast and stable algorithm for solving a class of optimization problems that arise in many statistical estimation procedures, such as *sparse fused lasso over a graph*, *convex clustering*, and *trend filtering*, among others. We propose a so-called augmented *alternating direction methods of multipliers* (ADMM) algorithm to solve this class of problems. Compared to a standard ADMM algorithm, our proposal significantly reduces the computational cost at each iteration while maintaining roughly the same overall convergence speed. We also consider a new varying penalty scheme for the ADMM algorithm, which could further accelerate the convergence, especially when solving a sequence of problems with tuning parameters of different scales. Extensive numerical experiments on the sparse fused lasso problem show that the proposed algorithm is more efficient than the standard ADMM and two other existing state-of-the-art specialized algorithms. Finally, we discuss a possible extension and some interesting connections to two well-known algorithms. Supplemental materials for the article are available online.

Keywords: alternating direction methods of multipliers, generalized lasso, large-scale nonsmooth convex optimization

[*]This research was supported by a start-up fund from The Ohio State University. Special thanks to Peter Craigmile for reading the manuscript and giving us invaluable comments on the work. The author thanks the editor, the associate editor, and two anonymous referees for constructive comments.

1 Introduction

Many statistical procedures rely on regularization (or penalization) to strike a proper balance between model bias and model variability. Classical regularization functions are mostly smooth, which is possibly due to its amenability to computation. Non-smooth regularization functions have attracted increasing attention in the past two decades. Its popularity is perhaps due to its ability to not only achieve a proper bias/variance trade-off, but it also allows the fitted model to exhibit certain desired structures. A notable example is the pursuit of sparsity structure through *lasso* regularization (Tibshirani, 1996; Chen et al., 1998).

On the other hand, the use of non-smooth regularization often creates more challenging optimization problems. In this article, we consider a particular type of non-smooth convex optimization problem that arises in many statistical estimation problems. Specifically, we aim to solve optimization problems of the following form

$$\underset{\theta \in \mathbb{R}^p}{\text{minimize}}\ f(\theta) + g(A\theta), \tag{1}$$

where both $f(\cdot)$ and $g(\cdot)$ are "simple" convex functions (in a sense to be made precise), and $A \in \mathbb{R}^{m\times p}$. Moreover, we allow $f(\cdot)$ and $g(\cdot)$ to be defined on an extended real line so that (1) also includes problems with constraints.

Consider, for example, the *generalized lasso* problem (Tibshirani and Taylor, 2011)

$$\underset{\beta \in \mathbb{R}^p}{\text{minimize}}\ (1/2) \cdot \|y - X\beta\|_2^2 + \lambda \|A\beta\|_1, \tag{2}$$

where $y \in \mathbb{R}^n$ is a response vector, $X \in \mathbb{R}^{n\times p}$ is a data matrix, λ is a tuning parameter, and A is a user-specified penalty matrix. This problem is clearly a special case of (1). When $A = I$, it reduces to the popular *lasso* problem. In fact, with different choices of A and X, the above problem includes many already well-known problems in statistics: *fused lasso* (Tibshirani et al., 2005), *grouping pursuit* (Shen and Huang, 2010; Zhu et al., 2013; Ke et al., 2015), and *trend filtering* (Kim et al., 2009; Tibshirani et al., 2014; Ramdas and Tibshirani, 2014), among others.

In this article, we propose to solve (1) using a so-called augmented *alternating direction methods of multipliers* (ADMM) algorithm. The ADMM algorithm has attracted a lot of attention recently in both statistics and machine learning communities, partly because of its flexibility in simplifying a wide range of optimization problems that arise in statistical machine learning applications, and partly because of its general convergence properties. It has also been proved to be quite flexible and versatile in dealing with many large-scale statistical estimation problems (see, e.g., Boyd et al., 2011). Here we apply the ADMM algorithm to (1) in a somewhat non-standard fashion, leading to a more general and potentially more efficient algorithm. The resulting algorithm is not new; it turns out to be closely related to two well-known algorithms used in imaging literature. Nevertheless, our ADMM-based derivation gives an unified view of both algorithms.

1.1 Existing literature

There is a huge literature on large-scale non-smooth convex optimization of form (1). Although the traditional primal-dual interior point method could still be quite effective for problems with certain special structures (Kim et al., 2009), more recent developments have been focusing on fast first-order methods. In particular, some recent proposals for solving (1) (or its special cases) often fall into three different categories: (i) path following (homotopy) algorithms (Shen and Huang, 2010; Tibshirani and Taylor, 2011; Zhou and Wu, 2014; Arnold and Tibshirani, 2015); (ii) fast first-order methods for optimizing so-called *composite functions* (Becker et al., 2011; Beck and Teboulle, 2012); and (iii) alternating direction methods of multipliers (Boyd et al., 2011; Ramdas and Tibshirani, 2014).

Path following algorithms for solving special cases of (1) have been studied recently in Shen and Huang (2010); Tibshirani and Taylor (2011); Zhou and Wu (2014); Arnold and Tibshirani (2015). In particular, Shen and Huang (2010); Tibshirani and Taylor (2011); Arnold and Tibshirani (2015) consider the *generalized lasso* problem, where $f(\theta)$ is quadratic and $g(\cdot)$ is ℓ_1 norm. Zhou and Wu (2014) proposes a path following algorithm for a more general problem. But the proposal requires solving an ODE, which makes it undesirable for high-dimensional problems.

General fast first-order methods (Nesterov, 2007; Tseng, 2008; Beck and Teboulle, 2009) have become quite popular for optimizing the composite function, which is a sum of a smooth and a non-smooth convex function. However, its effectiveness may diminish greatly when the non-smooth part is not a "simple" function. To overcome this difficulty, Becker et al. (2011); Beck and Teboulle (2012) employ the smoothing technique of Nesterov (2005) and solve the smoothed objective function by a fast first-order method. However, this extra smoothing step typically makes the convergence rate slower, and fairly complicated backtracking procedures have to be used to select the step size.

The ADMM algorithm provides an alternative way for solving large-scale non-smooth optimization problems. Unlike fast first-order algorithms, it does not require line search, which often makes its implementation easier. For instance, Wahlberg et al. (2012) use the ADMM algorithm to solve a *fused lasso* problem which is a special case of (2). Their proposal, however, crucially depends on a banded structure of the linear operator A, without which one still has to solve a $p \times p$ linear system, which can be computationally prohibitive for large-scale problems. More recently, Ramdas and Tibshirani (2014) propose a specialized ADMM algorithm for solving a trend filtering problem. But their proposal also depends on the banded structure of A, hence may be inappropriate for problems involving more general linear operators.

1.2 Our approach

Our approach for solving (1) is based on a special application of the ADMM algorithm. First, let us work through the standard ADMM approach for solving (1). The standard ADMM proceeds by first decoupling the two functions in (1) by introducing new equality constraints $A\theta = \gamma$. Then, the standard ADMM solves an equivalent formulation of (1)

$$\underset{\theta \in \mathbb{R}^p, \gamma \in \mathbb{R}^m}{\text{minimize}}\ f(\theta) + g(\gamma) \quad \text{subject to } A\theta - \gamma = 0 \tag{3}$$

by alternately updating the primal variables (θ, γ) and associated dual variable α:

$$\theta^{k+1} = \underset{\theta \in \mathbb{R}^p}{\arg\min} \left(f(\theta) + \frac{\rho}{2} \left\| A\theta - \gamma^k + \rho^{-1}\alpha^k \right\|_2^2 \right), \tag{4a}$$

$$\gamma^{k+1} = \underset{\gamma \in \mathbb{R}^m}{\arg\min} \left(g(\gamma) + \frac{\rho}{2} \left\| A\theta^{k+1} - \gamma + \rho^{-1}\alpha^k \right\|_2^2 \right), \tag{4b}$$

$$\alpha^{k+1} = \alpha^k + \rho(A\theta^{k+1} - \gamma^{k+1}). \tag{4c}$$

In the above expressions, $\rho > 0$ is a positive *penalty parameter*. This updating scheme has been shown to be equivalent to the *proximal point algorithm* (Eckstein and Bertsekas, 1992), and thus converges under very mild conditions on $f(\cdot)$ and $g(\cdot)$. Also see He and Yuan (2012) for a proof of $O(1/k)$ rate of convergence for the ADMM algorithms.

Sometimes, the θ-update in (4a) is difficult to carry out, especially when A is not a diagonal matrix. To get around this difficulty, we consider an "augmented" variable $(\gamma, \tilde{\gamma})$, where $\tilde{\gamma}$ relates to θ through $\tilde{\gamma} = (D - A^\top A)^{1/2}\theta$ with $D \in \mathbb{R}^{p\times p}$ satisfying $D \succeq A^\top A$. This gives us a new formulation of (1)

$$\underset{\theta,\tilde{\gamma}\in\mathbb{R}^p,\gamma\in\mathbb{R}^m}{\text{minimize}}\ f(\theta) + g(\gamma) \ \text{ subject to } \begin{pmatrix} A \\ (D - A^\top A)^{1/2} \end{pmatrix} \theta - \begin{pmatrix} \gamma \\ \tilde{\gamma} \end{pmatrix} = 0\,. \tag{5}$$

Note that the augmented variable $\tilde{\gamma}$ and the associated equality constraint are totally redundant. However, when applying the standard ADMM algorithm to this seemingly more complicated formulation, we obtain surprisingly simple updates

$$\theta^{k+1} = \underset{\theta\in\mathbb{R}^p}{\arg\min} \left(f(\theta) + (2\alpha^k - \alpha^{k-1})^\top A\theta + \frac{\rho}{2}(\theta - \theta^k)^\top D(\theta - \theta^k) \right), \tag{6a}$$

$$\gamma^{k+1} = \underset{\gamma\in\mathbb{R}^m}{\arg\min} \left(g(\gamma) + \frac{\rho}{2}\left\| A\theta^{k+1} - \gamma + \rho^{-1}\alpha^k \right\|_2^2 \right), \tag{6b}$$

$$\alpha^{k+1} = \alpha^k + \rho(A\theta^{k+1} - \gamma^{k+1})\,, \tag{6c}$$

which does not involve the augmented $\tilde{\gamma}$ at all. We defer a detailed derivation and the analysis of its convergence properties until later sections. Here we would like to point out that when $D = A^\top A$, the above algorithm reduces back to the standard ADMM algorithm; and for a general D, the new θ-update (6a) can be viewed as a one-step MM update for solving the original θ-update (4a).

The main advantage of the above augmented ADMM is its added flexibility for choosing D to simplify the θ-update (6a). Such simplification is indeed possible in many problems, and this is the primary reason why the proposed augmented ADMM algorithm could be more efficient than the standard ADMM. In fact, for problems in which the problem dimension greatly exceeds the sample size, the augmented ADMM runs much faster than the standard ADMM algorithm. Moreover, we demonstrate through runtime experiments that the proposed algorithm is also competitive with two existing state-of-the-art algorithms on the sparse fused lasso problem.

1.3 Outline

The rest of the paper is structured as follows. In Section 2 we give a detailed derivation of the augmented ADMM algorithm and an interesting extension, and in Section 3 we discuss its connections to two existing algorithms that are widely used in the imaging literature. In Section 4 we present an application to a sparse fused lasso problem. In Section 5 we apply the the proposed method to simulated datasets to demonstrate its superior computational efficiency. More discussion is provided in the last section.

2 Augmented ADMM algorithm

This section gives a detailed derivation of the augmented ADMM algorithm described in (6). First, we introduce a few useful notions and results in convex analysis that proved to be useful in the derivation of the proposed augmented ADMM algorithm. Second, we formally establish the equivalence of (6) to a standard ADMM algorithm applying to (5). Third, we propose a new varying penalty scheme. Finally, we present an application of the augmented ADMM to a more general ADMM problem involving two linear operators.

2.1 Some useful results from convex analysis

This subsection introduces some useful notions and a few basic analytical results that will help to structure and clarify the subsequent analysis. Most of the material here can be readily found (often in more general and abstract form) in textbooks such as (Rockafellar, 1997). Reader well-versed in convex analysis may skip this subsection.

First, let us introduce the notion of *conjugate function* and *Fenchel-Moreau Theorem*. For any function $\Omega(\cdot)$, its conjugate function is defined by $\Omega^*(v) := \sup_u \langle u, v\rangle - \Omega(u)$, which is convex. Moreover, the Fenchel-Moreau Theorem says that if $\Omega(\cdot)$ is a closed and proper convex function, its *biconjugate function* (conjugate of conjugate) equals to itself, that is, $\Omega^{**}(\cdot) = \Omega(\cdot)$.

Next, we introduce the concept of *proximal map* and *Moreau's identity*. For a given function $\Omega(\cdot)$, its proximal map is another function defined by

$$\operatorname{prox}_{\Omega(\cdot)}(u) = \arg\min_{v} \left(\Omega(v) + \frac{1}{2}\|u - v\|_2^2 \right). \tag{7}$$

The proximal map $\operatorname{prox}_{\Omega(\cdot)}(u)$ exists and is unique for all u if $\Omega(\cdot)$ is a closed and proper convex function. Moreau's identity relates the proximal map of $\Omega(\cdot)$ to that of its conjugate function $\Omega^*(\cdot)$. According to Moreau's identity,

$$u = \operatorname{prox}_{t\Omega(\cdot)}(u) + t\,\operatorname{prox}_{t^{-1}\Omega^*(\cdot)}(t^{-1}u) \tag{8}$$

for any $t > 0$. Hence, the proximal map of the conjugate $\Omega^*(\cdot)$ is as easy to compute as $\Omega(\cdot)$.

Lastly, we introduce the usual assumptions made in the literature for analyzing convergence of the ADMM algorithms (see Boyd et al., 2011, Section. 3.2). First, we define the Lagrangian of (3)

$$L(\theta, \gamma, \alpha) = f(\theta) + g(\gamma) + \alpha^\top (A\theta - \gamma), \tag{9}$$

where $\alpha \in \mathbb{R}^m$ is often referred to as the Lagrangian dual variable. Then, the dual problem to (3) equates to maximizing the dual function $\inf_{\theta,\gamma} L(\theta, \gamma, \alpha)$, or equivalently,

$$\underset{\alpha \in \mathbb{R}^m}{\text{maximize}}\ -f^*(-A^\top \alpha) - g^*(\alpha), \tag{10}$$

Throughout the rest of the section, we make the following assumptions about problem (3):

Assumption A:
(i) The functions $f : \mathbb{R}^p \to \mathbb{R} \cup \{+\infty\}$ and $g : \mathbb{R}^m \to \mathbb{R} \cup \{+\infty\}$ in (1) are closed, proper, and convex;
(ii) the Lagrangian function as defined in (9) has a saddle point (i.e., there exist $(\theta^\star, \gamma^\star, \alpha^\star)$ for which $L(\theta^\star, \gamma^\star, \alpha) \leq L(\theta^\star, \gamma^\star, \alpha^\star) \leq L(\theta, \gamma, \alpha^\star)$ holds for all θ, γ, α).

Note that the second assumption is equivalent to strong duality, which can be ensured by constraint qualifications such as Slater's condition (see Section 5.4.2 of Boyd et al., 2011).

2.2 Derivation of the augmented ADMM algorithm

Recall that our goal is to solve (1) and our augmented ADMM algorithm considers solving (5), which is an equivalent formulation of (1). Again, the only difference between the new formulation (5) and the standard ADMM formulation (3) is the newly introduced "augmented" variable $\tilde{\gamma} = (D - A^\top A)^{1/2}\theta$. The motivation behind choosing this particular equality constraint is to simplify the θ-update when we apply the standard ADMM to the new formulation (5).

Applying the standard ADMM algorithm to (5) by alternating between θ and $(\gamma, \tilde{\gamma})$ gives the following ADMM updates

$$\begin{aligned}
\theta^{k+1} &= \operatorname*{arg\,min}_{\theta \in \mathbb{R}^p} \Big(f(\theta) + \frac{\rho}{2}\big(\|A\theta - \gamma^k + \rho^{-1}\alpha^k\|_2^2 \\
&\qquad\qquad + \|(D - A^\top A)^{1/2}\theta - \tilde{\gamma}^k + \rho^{-1}\tilde{\alpha}^k\|_2^2 \big)\Big), && \text{(11a)} \\
\gamma^{k+1} &= \operatorname*{arg\,min}_{\gamma \in \mathbb{R}^m} \Big(g(\gamma) + \frac{\rho}{2}\|A\theta^{k+1} - \gamma + \rho^{-1}\alpha^k\|_2^2 \Big), && \text{(11b)} \\
\tilde{\gamma}^{k+1} &= (D - A^\top A)^{1/2}\theta^{k+1} + \rho^{-1}\tilde{\alpha}^k, && \text{(11c)} \\
\alpha^{k+1} &= \alpha^k + \rho(A\theta^{k+1} - \gamma^{k+1}), && \text{(11d)} \\
\tilde{\alpha}^{k+1} &= \tilde{\alpha}^k + \rho\left((D - A^\top A)^{1/2}\theta^{k+1} - \tilde{\gamma}^{k+1}\right), && \text{(11e)}
\end{aligned}$$

where $\alpha \in \mathbb{R}^m, \tilde{\alpha} \in \mathbb{R}^p$ are the associated dual variables. Basically, the augmented ADMM algorithm follows from the above updating scheme after some manipulation and simplification. First, combining (11c) and (11e) gives $\tilde{\alpha}^{k+1} = 0$. Then, substituting (11c) to (11a), the θ-update becomes

$$\theta^{k+1} = \operatorname*{arg\,min}_{\theta \in \mathbb{R}^p} \Big(f(\theta) + \frac{\rho}{2}\big(\|A\theta - \gamma^k + \rho^{-1}\alpha^k\|_2^2 + \|(D - A^\top A)^{1/2}(\theta - \theta^k)\|_2^2 \big)\Big), \qquad (12)$$

which, together with (11d), gives (6a). Now it becomes more clear as to why we add the additional constraint $(D - A^\top A)^{1/2}\theta = \tilde{\gamma}$ at the first place. This particular constraint simplifies the quadratic term $\theta A^\top A\theta$ in the θ-update (12).

Second, γ-update and α-update can be combined further and simplified. Indeed, combining (11b) and (11d) gives $\rho^{-1}\alpha^{k+1} = \rho^{-1}\alpha^k + A\theta^{k+1} - \operatorname{prox}_{\rho^{-1}g(\cdot)}\left(\rho^{-1}\alpha^k + A\theta^{k+1}\right)$. Then, Moreau's identity (8) allows us to further simplify it to $\alpha^{k+1} = \operatorname{prox}_{\rho g^*(\cdot)}\left(\alpha^k + \rho A\theta^{k+1}\right)$. As a result, there is no need to keep track of the newly introduced "augmented" variable $(\gamma, \tilde{\gamma})$; the augmented ADMM algorithm updates the primal variable θ and the dual variable α alternately. We summarize this augmented ADMM algorithm and its convergence result in the following theorem.

Theorem 1 *Under* ***Assumption A****, for any matrix $D \in \mathbb{R}^{p\times p}$ satisfying $D \succeq A^\top A$ and any positive scalar $\rho > 0$, the following update*

$$\theta^{k+1} = \arg\min_{\theta\in\mathbb{R}^p} \left(f(\theta) + (2\alpha^k - \alpha^{k-1})^\top A\theta + \frac{\rho}{2}(\theta - \theta^k)^\top D(\theta - \theta^k)) \right), \tag{13a}$$

$$\alpha^{k+1} = \text{prox}_{\rho g^*(\cdot)} \left(\alpha^k + \rho A\theta^{k+1}\right) \tag{13b}$$

converges in the sense that primal objective functions along the sequence of primal variables converge to the optimal value: $f(\theta^k) + g(A\theta^k) \to \inf_\theta f(\theta) + g(A\theta)$, and the sequence of dual variables converge to an optimal dual point: $\alpha^k \to \alpha^\star$, where $\alpha^\star$ is the optimal point of (10)*—the Lagrangian dual problem to* (1)*.*

The proof is straightforward given standard convergence results for ADMM algorithms (see Section 3.2 of Boyd et al., 2011) and the equivalence between the augmented ADMM and the standard ADMM that we just showed above. A more detailed proof is provided in the online Supplemental materials. Again, we emphasize that D can be any matrix that dominates $A^\top A$. In particular, if $D = A^\top A$, the above augmented ADMM algorithm reduces back to the standard ADMM algorithm

$$\theta^{k+1} = \arg\min_{\theta\in\mathbb{R}^p} \left(f(\theta) + (2\alpha^k - \alpha^{k-1})^\top A\theta + \frac{\rho}{2}(\theta - \theta^k)^\top A^\top A(\theta - \theta^k)) \right), \tag{14a}$$

$$\alpha^{k+1} = \text{prox}_{\rho g^*(\cdot)} \left(\alpha^k + \rho A\theta^{k+1}\right). \tag{14b}$$

As pointed out by one referee, the objective function in (13a) can be viewed as a majorization of that in (14a) at the current iterate θ^k, although it is unclear why such combination still preserves the algorithm's convergent property. Similar ideas have also been used in the literature to simplify the standard ADMM algorithm (Chen and Teboulle, 1994; Zhang et al., 2011, 2010; Esser et al., 2009), but they all require separate convergence analysis.

For specific problems, we would like to choose D to simplify the θ-update (13a) (as compared to (14a)), which is possible in many cases. For example, a simple choice would be $D = \delta I$ with $\delta \geq \|A\|_{\text{op}}^2$, where $\|A\|_{\text{op}}$ denotes the operator norm of A. With this choice, the θ-update (13a) can be viewed as a proximal map of $f(\cdot)$

$$\theta^{k+1} = \text{prox}_{(\rho\delta)^{-1}f} \left(\theta^k - (\rho\delta)^{-1}A^\top(2\alpha^k - \alpha^{k-1})\right) .$$

As a final remark, the α-update (13b) requires that the proximal map of the conjugate function of $g(\cdot)$ is easy to calculate. Fortunately, this is the case in many statistical estimation problems—the regularization function $g(\cdot)$ is simple enough that very efficient algorithms are available to calculate its proximal map (and hence that of its conjugate $g^*(\cdot)$ thanks to the Moreau's identity (8)).

2.3 Stopping criterion and a new varying penalty strategy

We use the standard stopping criterion for the ADMM algorithm based on primal and dual residuals following Section 3.3 of Boyd et al. (2011). Specifically, we define the primal and dual residuals

$$\begin{aligned} r^{k+1} &= A\theta^{k+1} - \gamma^{k+1} = \rho^{-1}(\alpha^{k+1} - \alpha^k), \\ s^{k+1} &= \rho A^\top(\gamma^{k+1} - \gamma^k) = \rho A^\top A(\theta^{k+1} - \theta^k) + A^\top(2\alpha^k - \alpha^{k-1} - \alpha^{k+1}) \end{aligned}$$

as well as the termination criterion $\|r^{k+1}\|_2 \leq \epsilon^{\text{pri}}$, $\|s^{k+1}\|_2 \leq \epsilon^{\text{dual}}$, where the primal and dual feasibility ϵ^{pri} and ϵ^{dual} are specified as suggested by Boyd et al. (2011).

Theoretically, the primal and dual residuals r^k and s^k will converge to zero for any fixed penalty parameter $\rho > 0$. In practice, however, the convergence speed depends heavily on the choice of ρ, and it is often not clear how to choose ρ, especially when solving a sequence of problems with different tuning parameters. A simple varying penalty strategy is suggested in Section 3.4.1 of Boyd et al. (2011). However, we find this strategy unstable, sometimes creating too many redundant ρ-updating steps. This instability has been pointed out recently by Ramdas and Tibshirani (2014).

To overcome this unstable behavior, we propose a slightly different varying penalty strategy with two minor modifications. First, we set the updating rule for ρ to be

$$\rho = \begin{cases} \eta\rho & \text{if } \|r^k\|_2/\epsilon^{\text{pri}} \geq \mu\|s^k\|_2/\epsilon^{\text{dual}} \\ \eta^{-1}\rho & \text{if } \|s^k\|_2/\epsilon^{\text{dual}} \geq \mu\|r^k\|_2/\epsilon^{\text{pri}}, \end{cases} \tag{15}$$

where η, μ are set to be 2 and 10 as suggested by Boyd et al. (2011). Note that the difference here is that the primal and dual residual are scaled further by the primal and dual feasibility parameter, respectively. The idea is to improve convergence when primal and dual feasibilities are on different scales.

Moreover, instead of invoking (15) at every iteration as suggested in Boyd et al. (2011), we only update ρ for every k_i iterations with $\{k_i\}_{i=1}^{\infty}$ being an increasing sequence and $\lim_{i\to\infty} k_i = \infty$. In other words, we only invoke updating rule (15) when $k = \sum_{i=1}^{l} k_i$ for some positive integer l. Ideally, we should choose the first few k_i's such that the complexity of k_i's ADMM iteration is roughly the same as that of performing a proximal map of f. On these grounds, we choose $k_i = i \min(n, p)$, which works quite well in practice.

Based on our limited experience, the resulting new varying penalty strategy is much more stable and greatly reduces the number of ρ-updating. Moreover, the convergence speed could be accelerated substantially by this new strategy, especially when solving a sequence of problems with different A's.

2.4 An extension

In this subsection, we apply the augmented ADMM algorithm to more general problems in which both parts in the objective function are composition of a linear operator and a simple function. This is motivated by statistical estimation problems in which the proximal map of $f(\cdot)$ may be difficult to evaluate, but $f(\cdot)$ is a composition of a linear operator and a "simple" function. For example, empirical loss functions for linear models are often a composition of a linear operator and a "simple" function. In such cases, we demonstrate that the augmented ADMM could still be quite useful. Specifically, consider the following problem

$$\underset{\theta\in\mathbb{R}^p}{\text{minimize}}\; g_1(A_1\theta) + g_2(A_2\theta),$$

where $A_1 \in \mathbb{R}^{m_1\times p}$ and $A_2 \in \mathbb{R}^{m_2\times p}$. Again we assume that both $g_1(\cdot)$ and $g_2(\cdot)$ are closed proper convex functions and their proximal maps are easy to calculate.

Although the above problem seems to be more general than (1), the augmented ADMM algorithm for (1) can still be applied here. Specifically, note that we can think of the entire objective function $g_1(A_1\theta) + g_2(A_2\theta)$ as $g(A\theta)$ in (1) with $A = \begin{pmatrix} A_1 \\ A_2 \end{pmatrix}$, and apply the augmented ADMM algorithm (13) to the above problem with $g(A\theta) = g_1(A_1\theta) + g_2(A_2\theta)$

and $f(\theta) = 0$. The resulting updating scheme is

$$\begin{aligned}
&\theta^{k+1} = \theta^k - \rho^{-1}D^{-1}\big(A_1^\top \bar{\alpha}_1^k + A_2^\top \bar{\alpha}_2^k\big),\\
&\alpha_1^{k+1} = \mathrm{prox}_{\rho g_1^*(\cdot)}\big(\rho A_1 \theta^{k+1} + \alpha_1^k\big), \quad \bar{\alpha}_1^{k+1} = 2\alpha_1^{k+1} - \alpha_1^k,\\
&\alpha_2^{k+1} = \mathrm{prox}_{\rho g_2^*(\cdot)}\big(\rho A_2 \theta^{k+1} + \alpha_2^k\big), \quad \bar{\alpha}_2^{k+1} = 2\alpha_2^{k+1} - \alpha_2^k\,,
\end{aligned}$$

where $D \succeq A_1^\top A_1 + A_2^\top A_2$. It is worth noting that we only need to perform one Cholesky factorization for D at the beginning and cache it for all subsequent iterations. This is in contrast to the augmented ADMM algorithm for (1), in which Cholesky factorization has to be recomputed every time we change ρ.

3 Connections to existing algorithms

In this section, we shall establish equivalence to two well known algorithms used in the image processing literature. Specifically, we show that the augmented ADMM can be understood as the first-order primal-dual algorithm (Chambolle and Pock, 2011) and the linearized ADMM algorithm (Chen and Teboulle, 1994; Zhang et al., 2011, 2010; Esser et al., 2009). Both algorithms have been extensively studied in the image processing literature. As a result, existing convergence theories for the ADMM algorithm would also apply to both algorithms, leading to a unifying theoretical framework.

3.1 First-order primal-dual algorithm

The first-order primal-dual algorithm proposed by Chambolle and Pock (2011) solves exactly the same problem (1) by primal-dual updates. Their algorithm is remarkably similar to the proposed augmented ADMM algorithm when D is an identity matrix.

Indeed, their main algorithm (c.f. **Algorithm 1** of Chambolle and Pock (2011)) is equivalent to the following updates

$$\begin{aligned}
\alpha^{k+1} &= \mathrm{prox}_{\sigma g^*(\cdot)}\big(\alpha^k + \sigma A \bar{\theta}^k\big)\\
\theta^{k+1} &= \mathrm{prox}_{\tau f(\cdot)}\big(\theta^k - \tau A^\top \alpha^{k+1}\big)\\
\bar{\theta}^{k+1} &= 2\theta^{k+1} - \theta^k\,,
\end{aligned} \tag{16}$$

where σ, τ are positive scalars. Convergence analysis is performed for the above updates under the condition that $\sigma\tau\|A\|_{\mathrm{op}}^2 < 1$ (c.f. **Theorem 1** of Chambolle and Pock (2011)).

Specifically, a convergence rate of $O\big(1/k\big)$ is established. Note that this is the same convergence rate for the ADMM algorithm (He and Yuan, 2012).

To see their connection, we choose $D = \delta I$ with $\delta = (\sigma\tau)^{-1}$, the penalty parameter $\rho = \sigma$ in (13a), and define $\bar{\alpha}^k = 2\alpha^k - \alpha^{k-1}$. Then the augmented ADMM algorithm (13) reduces to

$$\begin{aligned} \theta^{k+1} &= \operatorname{prox}_{\tau f(\cdot)}\left(\theta^k - \tau A^\top \bar{\alpha}^k\right) \\ \alpha^{k+1} &= \operatorname{prox}_{\sigma g^*(\cdot)}\left(\alpha^k + \sigma A\theta^{k+1}\right) \\ \bar{\alpha}^{k+1} &= 2\alpha^{k+1} - \alpha^k\,, \end{aligned}$$

which looks almost identical to the primal-dual first order algorithm in (16) except that the ADMM algorithm performs an extrapolation of α^k (to $2\alpha^k - \alpha^{k-1}$) whereas the first-order primal-dual algorithm performs an extrapolation of θ^k. Moreover, the condition for convergence matches—our augmented ADMM requires that $\delta \geq \|A\|_{\text{op}}^2$, which is equivalent to $\sigma\tau\|A\|_{\text{op}}^2 \leq 1$. Of course, this similarity is not a coincidence. The next theorem establishes their connections.

Theorem 2 *The first-order primal-dual algorithm (**Theorem 1** of Chambolle and Pock (2011)) is equivalent to the augmented ADMM algorithm* (13) *applied to the Lagrangian dual to* (1) *with $D = (\sigma\tau)^{-1}I$ and $\rho = \tau$.*

The proof is provided in the online Supplemental materials. In essence, the first-order primal-dual algorithm is equivalent to applying the augmented ADMM algorithm to the dual problem.

3.2 Linearized/preconditioned ADMM

Linearized/preconditioned ADMM (Chen and Teboulle, 1994; Esser et al., 2009; Zhang et al., 2010, 2011) is proposed with the same motivation—to simplify the θ-update in the standard ADMM algorithm. Although it is related to the ADMM algorithm, to the best of our knowledge, their equivalence has not been established in the literature.

The idea of the linearized ADMM is to modify the θ-update (4a) of the standard ADMM algorithm by adding an extra quadratic term $\frac{1}{2}(\theta - \theta^k)^\top Q(\theta - \theta^k)$ with $Q \succeq 0$.

This modification leads to the following θ-update:

$$\theta^{k+1} = \underset{\theta \in \mathbb{R}^p}{\arg\min} \left(f(\theta) + \frac{\rho}{2} \left\| A\theta - \gamma^k + \rho^{-1}\alpha^k \right\|_2^2 + \frac{1}{2}(\theta - \theta^k)^\top Q(\theta - \theta^k) \right).$$

It is easy to verify that it is equivalent to the augmented ADMM algorithm. Hence, in view of the augmented ADMM algorithm, the linearized/preconditioned ADMM algorithm is just ADMM algorithms that is applied to a new formulation. Thus, all the convergence results of the ADMM algorithm would apply to the the linearized/preconditioned ADMM.

4 Application to the sparse fused lasso over a graph

In this section, we consider the so-called sparse fused lasso problem. Interests in this type of problems often stem from statistical modeling of genomic data (Tibshirani et al., 2005; Bondell and Reich, 2008; Shen and Huang, 2010; Zhu et al., 2013; Ke et al., 2015) and fMRI data (Xin et al., 2014). The proposed augmented ADMM algorithm is readily applicable in this case, and is especially appealing when the problem dimension greatly exceeds the sample size. We also illustrate its advantage over the standard ADMM algorithm by comparing their computational complexities.

4.1 Problem setup

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph, where $\mathcal{V}$ is the node set and $\mathcal{E}$ is the edge set. Often the node set $\mathcal{V}$ encodes the features/covariates in the model and the edge set encodes their relationships. Based on such a graph, we consider the following optimization problem

$$\underset{\beta \in \mathbb{R}^p}{\text{minimize}}\, (1/2) \cdot \|y - X\beta\|_2^2 + \lambda \sum_{(i,j)\in\mathcal{E}} |\beta_i - \beta_j| + \nu \cdot \lambda \|\beta\|_1, \tag{17}$$

where $y \in \mathbb{R}^n$ is the response vector, $X \in \mathbb{R}^{n\times p}$ is a data matrix, and (λ, ν) are regularization parameters. This problem is commonly referred to as the sparse fused lasso over a graph. Usually one needs to solve many instances of (17) with different choices of tuning parameters (λ, ν).

Note that when $X = I$, the above problem can be solved very efficiently using specialized algorithm (see, e.g., Chambolle and Darbon, 2009). For a general X, the above problem becomes much more challenging.

4.2 ADMM algorithm

This subsection applies the augmented ADMM algorithm to (17). We let $\lambda_1 = \lambda\nu$ and $\lambda_2 = \lambda$ and write (17) in the form of (1) with $A = \begin{bmatrix} I \\ C \end{bmatrix}$ and $g(\gamma) = \lambda_1\|\gamma_1\|_1 + \lambda_2\|\gamma_2\|_1$ where C is the oriented incidence matrix associated with graph $\mathcal{G}$ and $\gamma = \begin{bmatrix} \gamma_1 \\ \gamma_2 \end{bmatrix}$ with $\gamma_1 \in \mathbb{R}^p, \gamma_2 \in \mathbb{R}^m$. Then, applying the augmented ADMM gives the following updates

$$\begin{aligned}
\beta^{k+1} &= \left(\rho D + X^\top X\right)^{-1}\left(\rho D\beta^k + X^\top y - A^\top(2\alpha^k - \alpha^{k-1})\right), \\
\alpha^{k+1} &= \mathcal{P}_{\{\|\alpha_1\|_\infty \le \lambda_1, \|\alpha_2\|_\infty \le \lambda_2\}}\left(\alpha^k + \rho A\beta^{k+1}\right),
\end{aligned}$$

where $\alpha = \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} \in \mathbb{R}^{p+m}$ with $\alpha_1 \in \mathbb{R}^p, \alpha_2 \in \mathbb{R}^m$, and $D \in \mathbb{R}^{p\times p}$ satisfying $D \succeq I + C^\top C$. In the augmented ADMM algorithm, we choose $D = \text{Diag}\Big(2d_1 + 1, \cdots, 2d_p + 1\Big)$, where d_i is the degree of the ith node in $\mathcal{G}$, $i = 1, \cdots, p$. On the other hand, choosing $D = A^\top A$ gives the standard ADMM updates.

Clearly, the difference lies in the β-updates: the β-update for the augmented ADMM algorithm involves inverting matrix $\rho D + X^\top X$, whereas the standard ADMM algorithm involves inverting $\rho A^\top A + X^\top X$. The difference in carrying out these matrix inversions would be huge when $p \gg n$. Next, we perform a detailed analysis of the computational complexity of both ADMM algorithms for solving (17).

4.3 Computational complexity

This subsection investigates the computational complexities of both ADMM algorithms for solving (17). We first introduce some notation to be used in our analysis. Let N_{admm} be the number of ADMM iterations and N_{chol} the total number of Cholesky factorizations. Let $a \vee b$ and $a \wedge b$ denote the larger and smaller number between a and b, respectively. We should stress at the outset that N_{chol} is typically less than 10, even when solving a path of solutions, and that N_{admm} typically scales as $O\big(1/\epsilon\big)$ based on convergence theory for the ADMM algorithm (see, e.g., He and Yuan, 2012).

We summarize the computational complexities for both ADMM algorithms in Table 1. Detailed derivations are provided in the online Supplemental materials. Note that the computational complexity for both algorithms is identical when $p \leq n$. However, when $p > n$, the computational complexity for the augmented ADMM is linear in p (if $m < np$), whereas the computational complexity for the standard ADMM is cubic in p for Cholesky factorization and quadratic in p for each ADMM iteration. The gain in computational efficiency is due to an application of the Woodbury matrix identity to invert a matrix that is a sum of a diagonal matrix and a low rank matrix (see the online Supplemental materials for more details). The gain in efficiency could be huge for targeted applications where $p \gg n$.

	$p \leq n$	$p > n$
augADMM	$O\left(N_{\text{chol}} p^2 n + N_{\text{admm}} p^2\right)$	$O\left(N_{\text{chol}} n^2 p + N_{\text{admm}} [pn \vee m]\right)$
stanADMM	$O\left(N_{\text{chol}} p^2 n + N_{\text{admm}} p^2\right)$	$O\left(N_{\text{chol}} p^3 + N_{\text{admm}} p^2\right)$

Table 1: Computational complexity of the augmented ADMM algorithm (augADMM) and the standard ADMM algorithm (stanADMM). Here N_{admm} and N_{chol} denote the number of ADMM iterations and the total number of Cholesky factorizations, respectively.

5 Numerical experiments

This section investigates the performance of the augmented ADMM algorithm (`augADMM`), the standard ADMM algorithm (`stanADMM`), a path following algorithm (`genlasso`) proposed by Arnold and Tibshirani (2015), and an accelerated proximal gradient method (`fGFL`) proposed by Xin et al. (2014). The later two algorithms were designed specifically for solving the sparse fused lasso problem. Note that the `fGFL` algorithms combines a parametric flow algorithm with the fast first-order method `FISTA` (Beck and Teboulle, 2009). This characteristic seems to make `fGFL` more suitable for solving (17) at a fixed single tuning parameter, because no trivial warm start strategy could be used when solving the problem over a sequence of tuning parameters. In contrast, the path-following `genlasso` algorithm is more suitable when solving the problem over a sequence of tuning parameters.

5.1 Experimental settings

We consider a high-dimensional sparse fused lasso over a graph problem motivated by a gene network example considered by Li and Li (2010), where an entire network consists of many subnetworks, each with one transcription factor (TF) and its 10 regulatory target genes; see Li and Li (2010) for a display of the network.

To mimic a regulatory relationship, the predictor matrix $X \in \mathbb{R}^{n\times p}$ is generated as follows. First, predictors corresponding to the TFs' are generated according to $\mathcal{N}(0,1)$. Then predictors of each target gene and the TF are constructed to have a bivariate normal distribution with correlation .7. Moreover, the target genes are independent conditional on the TF. The true regression coefficients are set to be

$$\beta^0 = \big(\underbrace{1,\ldots,1}_{11},\underbrace{-1,\ldots,-1}_{11},\underbrace{2,\ldots,2}_{11},\underbrace{-2,\ldots,-2}_{11},\underbrace{0,\ldots,0}_{p-44}\big)^\top,$$

and the response vector $y = X\beta^0 + \varepsilon$ with $\varepsilon_1,\cdots,\varepsilon_n \overset{\text{iid}}{\sim} N(0,\sigma_e^2)$ with error variance $\sigma_e^2 = .1$.

We generate the graph $\mathcal{G}$ as follows. First, we specify all sub-networks to be complete graphs, which amounts to 55×200 edges. Second, we randomly add some "erroneous" edges between nodes from different sub-networks.

Given data (y, X) and the graph $\mathcal{G}$ specified as above, we shall investigate the performance of `augADMM` (proposed method), `stanADMM`, `genlasso`, and `fGFL` on the sparse fused lasso problem (17) either for a fixed tuning parameter or for a sequence of tuning parameters. All experiments were performed using an Intel(R) Xeon Dual Eight Core CPU at 2.7GHz with 384GB memory. The `fGFL` was implemented in `MATLAB` and `C++`, and the other three algorithms were implemented in `R` and `C++`.

5.2 Running time comparison

This subsection compares the empirical running time of the four algorithms for solving (17). As discussed previously, we exclude `genlasso` in our comparison when solving (17) at a fixed tuning parameter, and we exclude `fGFL` when solving (17) over a sequence of tuning parameters.

First, we report the running time of `augADMM`, `stanADMM`, and `fGFL` when solving (17) at 20 different pairs of tuning parameters (λ, ν) under four different levels of suboptimality.

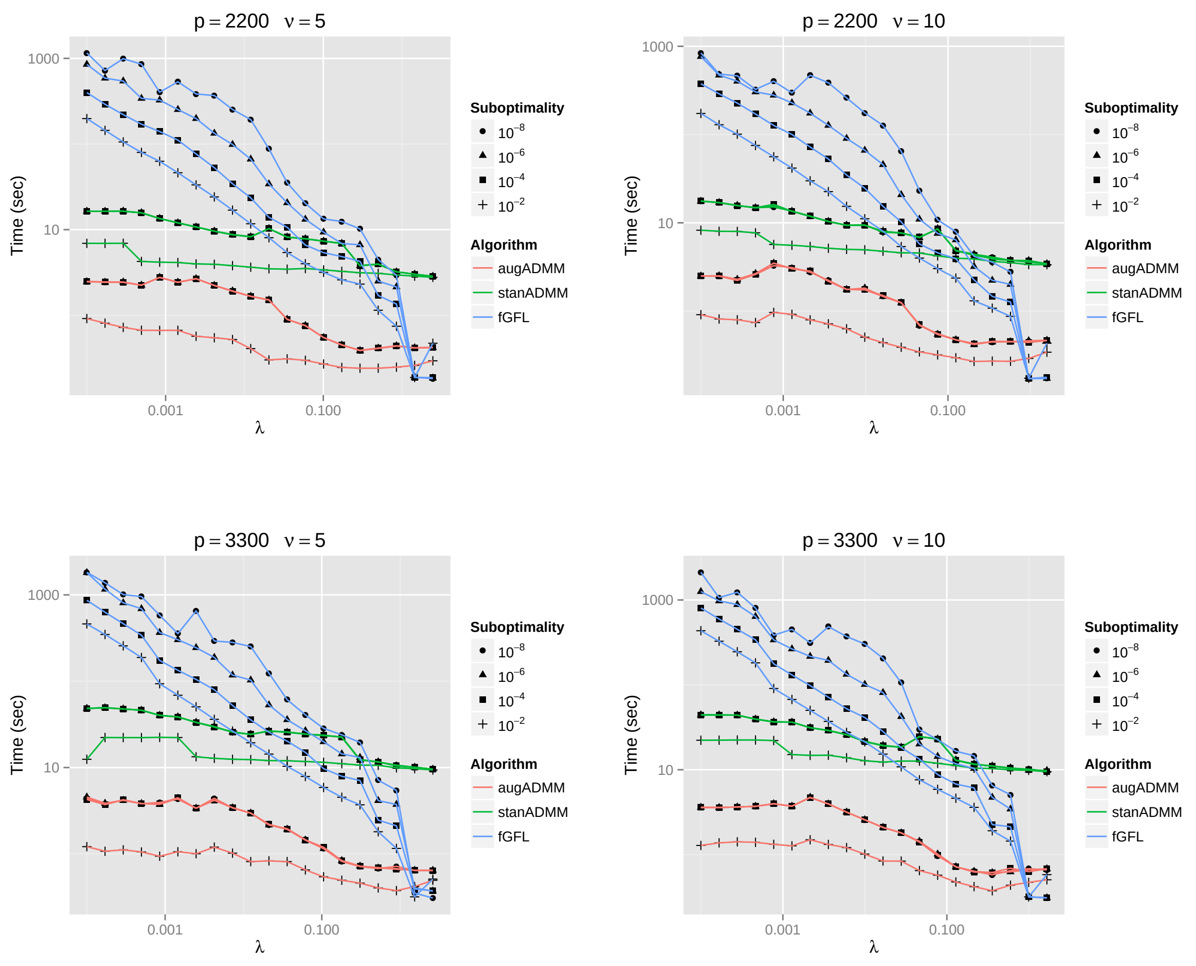

Figure 1: Runtime comparisons for the augmented ADMM (augADMM), the standard ADMM (stanADMM), and the fast proximal gradient method of Xin et al. (2014) (fGFL) under four different suboptimality: $10^{-2}, 10^{-4}, 10^{-6}, 10^{-8}$.

Specifically, for a fixed ν, we pick 20 different λ uniformly in log-scale from $[10^{-4}, \lambda_{\max}]$, where $\lambda_{\max}$ is the largest λ at which the solution path changes slope (calculated using `genlasso`). For each pair of (λ, ν), we run all algorithms until the relative function suboptimality reaches some desired level and we record their runtime. Here, we choose four different levels of suboptimality: $10^{-2}, 10^{-4}, 10^{-6}, 10^{-8}$. Note that the same starting point is used for problems with different tuning parameters.

The runtime results are shown in Figure 1. The four plots correspond to four different scenarios with different choices of (p, ν). Each plot shows the runtime of the three algorithms at 20 different values of λ under four levels of suboptimality. As see from Figure 1, the augmented ADMM (`augADMM`) runs the fastest for most choices of λ across all scenarios. Interestingly, the accelerated proximal gradient algorithm `fGFL` runs faster when λ was large, but it slows down significantly for smaller values of λ. The standard ADMM is less efficient than the augmented ADMM as expected, but it is still more efficient than `fGFL` for moderate and small values of λ. Also worthy of note is that the runtime for both ADMM algorithms to reach higher-accuracy $(10^{-4}, 10^{-6}, 10^{-8})$ solutions are essentially the same. This is due to the fact that the per-iteration cost of both ADMM algorithms are ignorable as compared to the Cholesky factorization steps, which often occur before the iterates attain to high accuracy.

Next we inspect the convergence speed of the three methods and how the operator norm of D influences the convergence speed of the augmented ADMM. Toward this end, we include three additional augmented ADMM algorithms using three different D's: (i) `augADMMx1` with $D = \|A\|_{\text{opt}}^2 I$; (ii) `augADMMx5` with $D = 5\|A\|_{\text{opt}}^2 I$; and (iii) `augADMMx10` with $D = 10\|A\|_{\text{opt}}^2 I$, where $\|A\|_{\text{opt}}$ denotes the operator norm of A. Then, we chooe four different pairs of tuning parameters: $\nu = 5$ or 10 and $\lambda = .01$ or .1, and plotted the objective suboptimality versus the number of iterations in Figure 2. Clearly `fGFL` converges faster than all the ADMM algorithms as expected, because the accelerated first order methods converge at rate of $O(1/k^2)$ whereas ADMM algorithms converge at rate of $O(1/k)$. Moreover, `stanADMM`, `augADMM`, and `augADMMx1` all have similar rates of convergence, although `stanADMM` usually converges slightly faster. Finally, increasing the operator norm of D slows down the convergence speed of augmented ADMM algorithm. Based on our limited experience, the number of iterations required (to reach a given suboptimality) would be at worst linearly dependent on the operator norm of D. Nevertheless, note that there are cases (see the second row of Figure 2) where increasing the operator norm of D does not change the convergence speed significantly.

From Figure 1 and Figure 2, we can see that although `fGFL` converges faster than the augmented ADMM algorithm, it still runs much slower, possibly due to its much higher per-iteration cost. Overall, the proposed augmented ADMM is the most efficient algorithm among the three competing algorithms for a wide range of tuning parameter values.

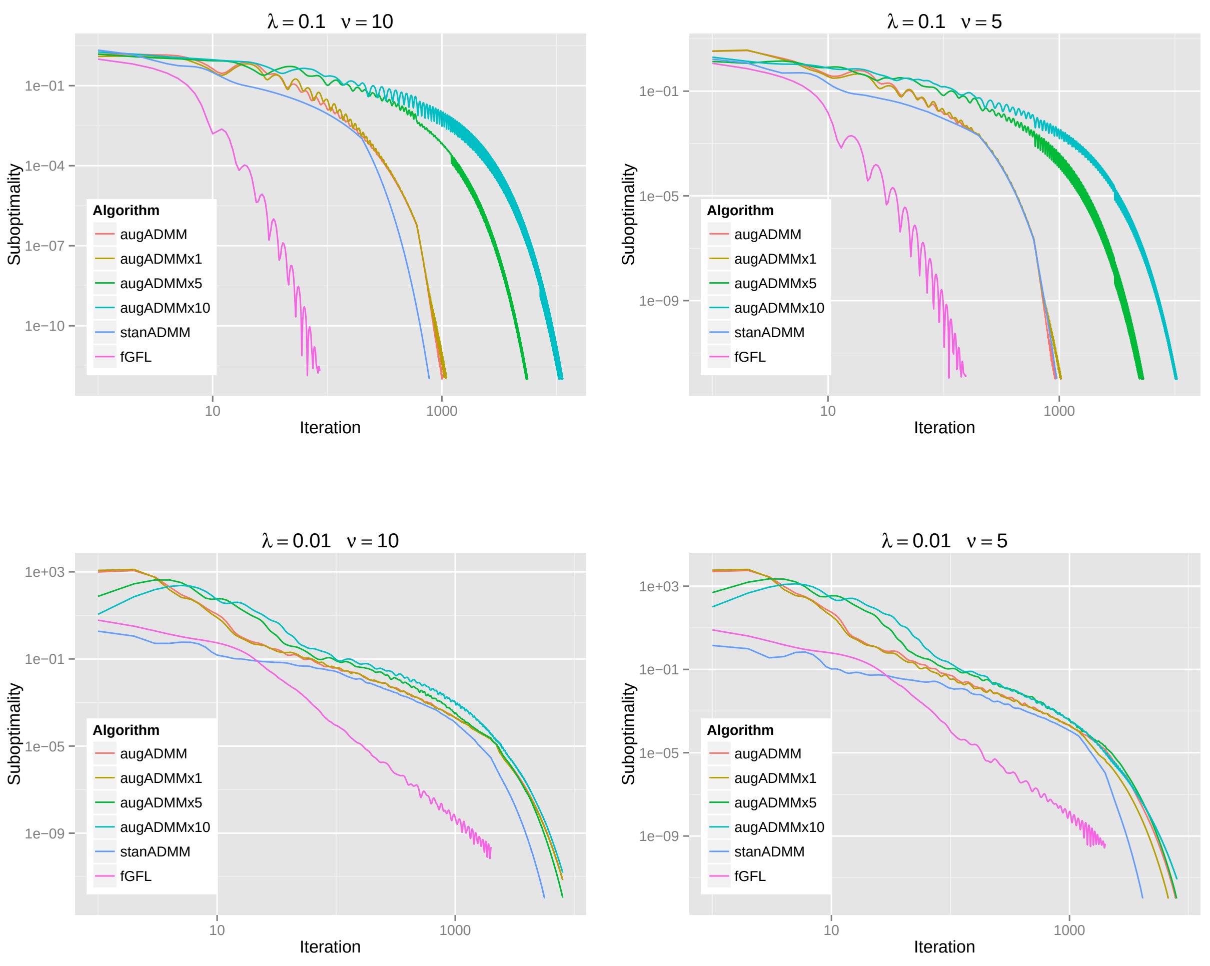

Figure 2: Convergence speed of the augmented ADMM (augADMM) with different choices of D, the standard ADMM (stanADMM), and the fast proximal gradient method of Xin et al. (2014) (fGFL).

Finally, we compare the runtime of `augADMM`, `stanADMM`, and `genlasso` when solving (17) at a sequence of different tuning parameters. Note that `genlasso` only approximately solves the problem when $p > n$. To make a fair comparison, we use the following parameter settings for the three algorithms. For the `genlasso` package, we change the default value of `minlam`—the value at which the algorithm terminates—from `0` to `1e-4`, because the default choice would make the algorithm extremely slow when the problem dimension is large. Moreover, we change the default value of `maxsteps`—the maximum number of steps—from `2000` to `1e5` so that it always compute the entire solution path. For both ADMM algorithms, we choose 100 tuning parameters (λ) uniformly in log-scale over $[10^{-4}, \lambda_{\max}]$. Stopping criteria are chosen such that both ADMM algorithms' suboptimality are comparable to each other and are smaller than that of `genlasso`.

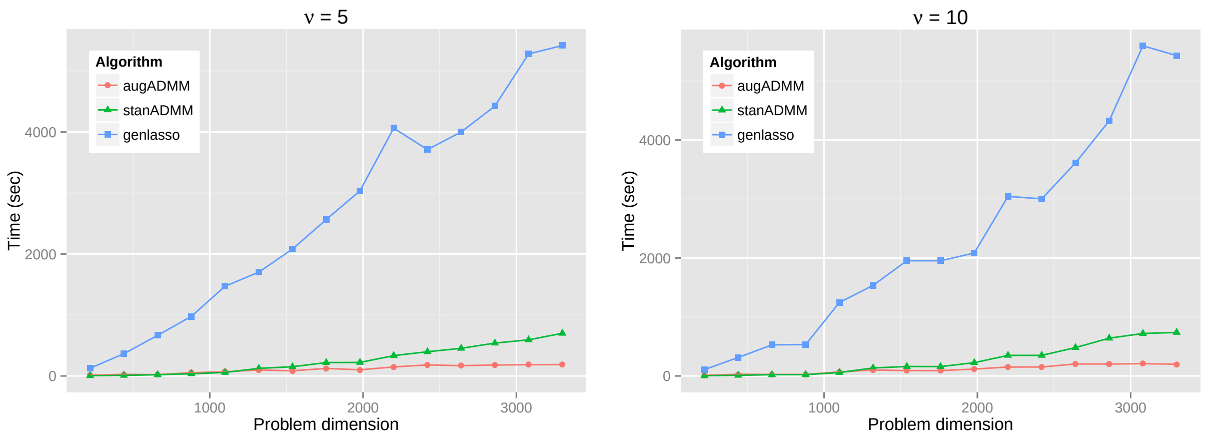

Figure 3: Runtime of the augmented ADMM (augADMM), the standard ADMM (stanADMM), and the path-following algorithm of Arnold and Tibshirani (2015) (genlasso) for finding a solution path.

With the above setup, we apply all algorithms to a sequence of problems with varying dimensions at two choices of ν, and plot their runtime in Figure 3. First, the augmented ADMM is comparable to the standard ADMM in problems with smaller dimensions, but it runs much faster for higher dimensions, which is in agreement with our earlier complexity analysis. Moreover, both ADMM algorithms run much faster than `genlasso`. However, readers should treat the comparison to `genlasso` with caution. On the one hand, `genlasso` computes the entire solution path over the selected tuning parameter range, whereas both ADMM algorithms only compute 100 solutions. On the other hand, `genlasso` has its own limitations: (i) it only approximately solves the problem when $p > n$; and (ii) it seems nontrivial to make it applicable to other loss functions other than the quadratic loss functions.

In summary, the augmented ADMM algorithm seems to be the algorithm of choice for solving the sparse fused lasso problem, especially when the number of covariates greatly exceeds the sample size.

6 Conclusions and future works

We investigated an augmented ADMM algorithm for solving a class of statistical estimation problem. We have shown that by introducing "augmented" variables, the resulting ADMM updates could be carried out more efficiently. Unlike the standard ADMM where one of the updates may involve a linear operator A, our augmented ADMM algorithm only requires evaluations of two proximal maps at each iteration. This leads to an algorithm with same (theoretical) convergence speed as that of the standard ADMM, but with much lower per-iteration cost in many situations. The efficiency of the augmented ADMM algorithm was demonstrated using a high-dimensional sparse fused lasso problem.

Although we mainly focused on a high-dimensional linear regression problem as an application of the proposed algorithm, many other potential applications such as, convex clustering (Hocking et al., 2011; Lindsten et al., 2011; Pan et al., 2013; Chi and Lange, 2014), trend filtering (Kim et al., 2009; Tibshirani et al., 2014; Ramdas and Tibshirani, 2014), and isotonic regression (Geyer, 1991), could also be considered. Moreover, for generalized linear models with the (sparse) fused lasso penalty, the extension introduced in Section 2.4 could be employed.

Finally, it is unclear how to choose D in (13). As suggested by one referee, one could let D be a diagonal matrix with diagonal elements $D_{ii} = \sum_{i=j}^{p} |(A^\top A)_{ij}|$ to ensure that D dominates $A^\top A$ (see Corollary 5.6.17 of Horn and Johnson, 2013). But this approach requires computing $A^\top A$, which might have high computational cost when the dimension of A is large. Another possible way could be to choose $D = c^2 I$, where c is an upper bound on the largest singular value of A. Simple upper bounds such as the one proposed by Byrne (2009) could be employed. Both approaches are worthy of further investigation.

Supplemental materials

Source code: The supplemental files for this article include R and MATLAB programs which can be used to reproduce the simulation study included in the article. Please read file README contained in the zip file for more details. (Zhu.zip, zip archive)

Appendix: The supplemental files include the Appendix which gives the proofs for Theorem 1 and 2, and a detailed analysis of the computational complexity for both ADMM algorithms. (ZhuAppendix.pdf)

References

Arnold, T. B. and R. J. Tibshirani (2015). Efficient implementations of the generalized lasso dual path algorithm. *Journal of Computational and Graphical Statistics* (just-accepted).

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

Beck, A. and M. Teboulle (2012). Smoothing and first order methods: A unified framework. *SIAM Journal on Optimization 22*(2), 557–580.

Becker, S. R., E. J. Candès, and M. C. Grant (2011). Templates for convex cone problems with applications to sparse signal recovery. *Mathematical Programming Computation 3*(3), 165–218.

Bondell, H. D. and B. J. Reich (2008). Simultaneous regression shrinkage, variable selection, and supervised clustering of predictors with oscar. *Biometrics 64*(1), 115–123.

Boyd, S., N. Parikh, E. Chu, B. Peleato, and J. Eckstein (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning 3*(1), 1–122.

Byrne, C. (2009). Bounds on the largest singular value of a matrix and the convergence of simultaneous and block-iterative algorithms for sparse linear systems. *International Transactions in Operational Research 16*(4), 465–479.

Chambolle, A. and J. Darbon (2009). On total variation minimization and surface evolution using parametric maximum flows. *International journal of computer vision 84*(3), 288–307.

Chambolle, A. and T. Pock (2011). A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision 40*(1), 120–145.

Chen, G. and M. Teboulle (1994). A proximal-based decomposition method for convex minimization problems. *Mathematical Programming 64*(1-3), 81–101.

Chen, S. S., D. L. Donoho, and M. A. Saunders (1998). Atomic decomposition by basis pursuit. *SIAM journal on scientific computing 20*(1), 33–61.

Chi, E. C. and K. Lange (2014). Splitting methods for convex clustering. *Journal of Computational and Graphical Statistics* (just-accepted), 00–00.

Eckstein, J. and D. P. Bertsekas (1992). On the douglasrachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming 55*(1-3), 293–318.

Esser, E., X. Zhang, and T. Chan (2009). A general framework for a class of first order primal-dual algorithms for tv minimization. *UCLA CAM Report*, 09–67.

Geyer, C. J. (1991). Constrained maximum likelihood exemplified by isotonic convex logistic regression. *Journal of the American Statistical Association 86*(415), 717–724.

He, B. and X. Yuan (2012). On non-ergodic convergence rate of douglas–rachford alternating direction method of multipliers. *Numerische Mathematik 130*(3), 567–577.

Hocking, T. D., A. Joulin, F. Bach, J.-P. Vert, et al. (2011). Clusterpath an algorithm for clustering using convex fusion penalties. In *28th international conference on machine learning*.

Horn, R. A. and C. R. Johnson (2013). *Matrix analysis*. Cambridge university press.

Ke, Z. T., J. Fan, and Y. Wu (2015). Homogeneity pursuit. *Journal of the American Statistical Association 110*(509), 175–194.

Kim, S.-J., K. Koh, S. Boyd, and D. Gorinevsky (2009). ℓ_1 trend filtering. *Siam Review 51*(2), 339–360.

Li, C. and H. Li (2010). Variable selection and regression analysis for graph-structured covariates with an application to genomics. *The Annals of Applied Statistics 4*(3), 1498.

Lindsten, F., H. Ohlsson, and L. Ljung (2011). Clustering using sum-of-norms regularization: With application to particle filter output computation. In *Statistical Signal Processing Workshop (SSP), 2011 IEEE*, pp. 201–204. IEEE.

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. *Technical report, CORE DISCUSSION PAPER*.

Pan, W., X. Shen, and B. Liu (2013). Cluster analysis: Unsupervised learning via supervised learning with a non-convex penalty. *Journal of Machine Learning Research 14*, 1865–1889.

Ramdas, A. and R. J. Tibshirani (2014). Fast and flexible admm algorithms for trend filtering. *arXiv preprint arXiv:1406.2082*.

Rockafellar, R. T. (1997). *Convex analysis*. Number 28. Princeton university press.

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

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

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

Tibshirani, R. and J. Taylor (2011, 06). The solution path of the generalized lasso. *The Annals of Statistics 39*(3), 1335–1371.

Tibshirani, R. J. et al. (2014). Adaptive piecewise polynomial estimation via trend filtering. *The Annals of Statistics 42*(1), 285–323.

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

Wahlberg, B., S. Boyd, M. Annergren, and Y. Wang (2012, July). An admm algorithm for a class of total variation regularized estimation problems. In *Proceedings 16th IFAC Symposium on System Identification*, Volume 16.

Xin, B., Y. Kawahara, Y. Wang, and W. Gao (2014). Efficient generalized fused lasso and its application to the diagnosis of alzheimers disease. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, pp. 2163–2169.

Zhang, X., M. Burger, X. Bresson, and S. Osher (2010). Bregmanized nonlocal regularization for deconvolution and sparse reconstruction. *SIAM Journal on Imaging Sciences 3*(3), 253–276.

Zhang, X., M. Burger, and S. Osher (2011). A unified primal-dual algorithm framework based on bregman iteration. *Journal of Scientific Computing 46*(1), 20–46.

Zhou, H. and Y. Wu (2014). A generic path algorithm for regularized statistical estimation. *Journal of the American Statistical Association 109*(506), 686–699.

Zhu, Y., X. Shen, and W. Pan (2013). Simultaneous grouping pursuit and feature selection over an undirected graph. *Journal of the American Statistical Association 108*(502), 713–725.