Alternating Proximal Gradient Method for Convex Minimization

Shiqian Ma*

September 09, 2012

Abstract. In this paper, we propose an alternating proximal gradient method that solves convex minimization problems with three or more separable blocks in the objective function. Our method is based on the framework of alternating direction method of multipliers. The main computational effort in each iteration of the proposed method is to compute the proximal mappings of the involved convex functions. The global convergence result of the proposed method is established. We show that many interesting problems arising from machine learning, statistics, medical imaging and computer vision can be solved by the proposed method. Numerical results on problems such as latent variable graphical model selection, stable principal component pursuit and compressive principal component pursuit are presented.

Key words. Alternating Direction Method of Multipliers, Proximal Gradient Method, Global Convergence, Sparse and Low-Rank Optimization

AMS subject classifications. 65K05, 90C25, 49M27

1. Introduction. In this paper, we consider solving the following separable convex optimization problem with linear linking constraint:

$$\begin{array}{ll} \min_{x\in\mathbb{R}^n, y\in\mathbb{R}^p} & f(x)+g(y) \\ \text{s.t.} & Ax+By=b, \end{array} \tag{1.1}$$

where $A\in\mathbb{R}^{m\times n}$, $B\in\mathbb{R}^{m\times p}$, $b\in\mathbb{R}^m$, and $f:\mathbb{R}^n\to\mathbb{R}$ and $g:\mathbb{R}^p\to\mathbb{R}$ are both convex functions. Furthermore, we assume that the proximal mappings of f and g are easy to obtain. By "easy", we usually mean that the proximal mapping can be obtained in an analytical form or can be computed in a computational effort that is comparable to computing a gradient or subgradient of the function. The proximal mapping of the function $h:\mathbb{R}^n\to\mathbb{R}$ for given $\xi>0$ and $z\in\mathbb{R}^n$ is defined as:

$$\mathrm{Prox}(h,\xi,z) := \mathrm{argmin}_x\ h(x)+\frac{1}{2\xi}\|x-z\|_2^2. \tag{1.2}$$

Functions with easy proximal mappings include ℓ_1 norm of a vector $\|x\|_1:=\sum_i|x_i|$, Euclidean norm of vector x (denoted as $\|x\|_2$), nuclear norm of a matrix X (denoted as $\|X\|_*$), which is defined as the sum of singular values of X, etc.

(1.1) is usually referred to as convex problem with two blocks, because there are two blocks of variables (x and y) and two convex functions (f and g) involved. The main purpose of this paper is to propose an alternating proximal gradient method for solving (1.1) that takes advantage of the structures of the functions, i.e., "easy" proximal mappings. The algorithm can be applied to solving problems with three or more separable blocks of variable and objective function

$$\begin{array}{ll} \min & \sum_{j=1}^N f_j(x_j) \\ \text{s.t.} & \sum_{j=1}^N A_jx_j=b, \end{array} \tag{1.3}$$

where $N\geq 3$, $f_j, j=1,\ldots,N$ are convex functions, $A_j\in\mathbb{R}^{m\times n_j}$, $x_j\in\mathbb{R}^{n_j}$ and $b\in\mathbb{R}^m$. In fact, the

*Institute for Mathematics and Its Applications, 400 Lind Hall, 207 Church Street SE, University of Minnesota, Minneapolis, MN 55455, USA. Email: maxxa007@ima.umn.edu.

general problem (1.3) can be reduced to the form of (1.1) by grouping the variables $x_1, \ldots, x_N$ into two blocks x and y. For example, we can rewrite (1.3) as the form of (1.1) by letting $x = x_1$, $y = [x_2^\top, \ldots, x_N^\top]^\top$, $f(x) = f_1(x_1)$, $g(y) = \sum_{j=2}^N f_j(x_j)$, $A = A_1$ and $B = [A_2, \ldots, A_N]$. Note that the proximal mapping of $g(y)$ consists of computing the $N-1$ proximal mappings of $f_2, \ldots, f_N$. In Section 3.3 we will discuss how to group the variables into two blocks such that it can be solved by our proposed method effectively.

Our proposed method for solving (1.1) uses only the first-order information. In each iteration, only computing the proximal mappings of f and g and the matrix vector multiplications Ax, $A^\top w$, By, $B^\top z$ are involved, and no matrix inversion or solving linear systems is required.

Convex optimization problems (1.1) and (1.3) have drawn a lot of attentions recently due to their emerging applications in sparse and low-rank optimization problems. Sparse and low-rank optimization problems concern problems with sparse or low-rank structures in their solutions. Many problems of this type arise from compressed sensing [7, 16] and its extensions and applications in sparse MRI [37], low-rank matrix completion [46, 6, 32, 8], sparse data fitting [50], sparse principal component analysis [62, 14], sparse graphical model selection [58, 1, 21], robust principal component pursuit [5, 10] etc.

One classical way to solve (1.1) is the augmented Lagrangian method

$$\left\{\begin{array}{lcl} (x^{k+1}, y^{k+1}) & := & \mathrm{argmin}_{x,y}\ \mathcal{L}_\mu(x, y; \lambda^k) \\ \lambda^{k+1} & := & \lambda^k - (Ax^{k+1} + By^{k+1} - b)/\mu, \end{array}\right. \tag{1.4}$$

where the augmented Lagrangian function $\mathcal{L}_\mu(x, y; \lambda)$ is defined as

$$\mathcal{L}_\mu(x, y; \lambda) := f(x) + g(y) - \langle \lambda, Ax + By - b\rangle + \frac{1}{2\mu}\|Ax + By - b\|_2^2, \tag{1.5}$$

in which λ is the Lagrange multiplier and $\mu > 0$ is a penalty parameter. However, minimizing the augmented Lagrangian function with respect to x and y simultaneously could be expensive in practice. Recently, the alternating direction method of multipliers (ADMM) has been studied extensively for solving sparse and low-rank optimization problems in the form of (1.1) and (1.3). A typical iteration of ADMM for solving (1.1) can be described as

$$\left\{\begin{array}{lcl} x^{k+1} & := & \mathrm{argmin}_x\ \mathcal{L}_\mu(x, y^k; \lambda^k) \\ y^{k+1} & := & \mathrm{argmin}_y\ \mathcal{L}_\mu(x^{k+1}, y; \lambda^k) \\ \lambda^{k+1} & := & \lambda^k - (Ax^{k+1} + By^{k+1} - b)/\mu. \end{array}\right. \tag{1.6}$$

A typical iteration of ADMM for solving (1.3) can be described as

$$\left\{\begin{array}{lcl} x_j^{k+1} & := & \mathrm{argmin}_{x_j}\mathcal{L}_\mu(x_1^{k+1}, \ldots, x_{j-1}^{k+1}, x_j, x_{j+1}^k, \ldots, x_N^k; \lambda^k), j = 1, \ldots, N \\ \lambda^{k+1} & := & \lambda^k - (\sum_{j=1}^N A_j x_j - b)/\mu, \end{array}\right. \tag{1.7}$$

where the augmented Lagrangian function $\mathcal{L}_\mu(x_1, \ldots, x_N; \lambda)$ is defined as

$$\mathcal{L}_\mu(x_1, \ldots, x_N; \lambda) := \sum_{j=1}^N f_j(x_j) - \langle \lambda, \sum_{j=1}^N A_j x_j - b\rangle + \frac{1}{2\mu}\|\sum_{j=1}^N A_j x_j - b\|^2, \tag{1.8}$$

in which λ is the Lagrange multiplier and $\mu > 0$ is a penalty parameter.

(1.6) is known as ADMM for solving problems with two blocks. This method is closely related to the

Douglas-Rachford operator splitting method (see e.g., [20, 18]), and the global convergence result has been well established. Also, there has been some recent progress on proving the convergence rate of (1.6) (see e.g., [26, 24, 30, 29, 2, 15, 27]). The problem with (1.6) is that the subproblems are easy only when both A and B are identity matrices. Note that when A and B are both identity matrices, the solutions of the two subproblems in (1.6) correspond to the proximal mappings of f and g, respectively. While when A (resp. B) is not an identity matrix, the first (resp. second) subproblem is usually not easy to solve and an iterative solver is needed to solve it, and this may become costly for solving a subproblem. (1.7) is known as ADMM for solving problems with three or more blocks when $N \geq 3$. The problem with (1.7) is that there was no global convergence result. Until very recently, Luo [36] showed the global convergence and linear convergence rate of (1.7) under certain assumptions. However, the assumptions in [36] are very strong so that many interesting applications arising from sparse and low-rank optimization do not satisfy these assumptions. Also, the subproblems in (1.7) are easy to solve only when $A_j, j = 1, \ldots, N$ are identity matrices.

Our contributions. We propose an alternating proximal gradient method (APGM) that solves (1.1). The more general problem (1.3) can also be solved by our APGM after we reduce it to the form of (1.1) by grouping the variables into two blocks. There are several advantages of our method. i) All the subproblems are easy. Each iteration of our proposed method only involves computing the N proximal mappings of $f_j, j = 1, \ldots, N$ and computing the matrix vector multiplications of matrices $A_j, A_j^\top, j = 1, \ldots, N$; ii) it only uses the first-order information and does not require any matrix inversion or solving linear systems; iii) we can prove that our proposed method globally converges to an optimal solution of (1.1) from any starting point; iv) we tested our methods on many interesting sparse and low-rank optimization problems from practice and the results showed that our method worked very well in practice and is suitable for large-scale problems. *The message we want to send from this paper is that sparse and low-rank optimization problems with multiple blocks and general linear linking constraints are solvable by alternating direction based methods with easy subproblems, and the global convergence is guaranteed.*

Organization. The rest of this paper is organized as follows. In Section 2 we discuss two interesting applications that are solvable by ADMM, and discuss a number of modern applications arising from machine learning, statistics, medical imaging and computer vision that are not readily solvable by ADMM. In Section 3, we propose our alternating proximal gradient method for solving (1.1) and establish its global convergence result. We show how to apply the proposed APGM to solving some modern applications in Section 4. In Section 5, we compare the performance of APGM and the linearized augmented Lagrangian method on problems from three different applications, and show that APGM is usually much faster. Finally, we draw some conclusion in Section 6.

2. Alternating direction method of multipliers. The ADMM (1.6) is closely related to the Douglas-Rachford and Peaceman-Rachford operator-splitting methods for finding zero of the sum of two maximal monotone operators. Douglas-Rachford and Peaceman-Rachford methods have been studied extensively in [17, 43, 35, 18, 20, 12, 13]. ADMM (1.6) has been revisited recently due to its success in the emerging applications of structured convex optimization problems arising from image processing, compressed sensing, machine learning, semidefinite programming and statistics etc. (see e.g., [23, 22, 52, 56, 28, 45, 59, 47, 24, 26, 41, 53, 3, 42, 38]).

2.1. Problems with two blocks. In this subsection, we discuss two problems that are suitable to ADMM with two blocks (1.6). Note that as we discussed in Section 1, the subproblems of ADMM (1.6) are easy to solve only when A and B are identity matrices. There are two problems with two separable blocks,

namely the sparse inverse covariance selection (SICS) and the robust principal component analysis (RPCA), that satisfy this requirement, i.e., A and B are both identity matrices.

SICS considers to estimate a sparse inverse covariance matrix of a multivariate Gaussian distribution from sample data. Let $X = \{x^{(1)}, \ldots, x^{(n)}\}$ be an n-dimensional random vector following an n-variate Gaussian distribution $\mathcal{N}(\mu, \Sigma)$, and let $G = (V, E)$ be a Markov network representing the conditional independence structure of $\mathcal{N}(\mu, \Sigma)$. Specifically, the set of vertices $V = \{1, \ldots, n\}$ corresponds to the set of variables in X, and the edge set E contains an edge (i, j) if and only if $x^{(i)}$ is conditionally dependent on $x^{(j)}$ given all remaining variables; i.e., the lack of an edge between i and j denotes the conditional independence of $x^{(i)}$ and $x^{(j)}$, which corresponds to a zero entry in the inverse covariance matrix Σ^{-1} ([33]). Thus learning the structure of this graphical model is equivalent to the problem of learning the zero-pattern of Σ^{-1}. The following convex formulation for estimating this sparse inverse covariance matrix has been suggested by [58, 1, 21]:

$$\min_{S \in \mathcal{S}_+^n} \ \langle \hat{\Sigma}, S \rangle - \log\det(S) + \rho \|S\|_1, \tag{2.1}$$

where $\mathcal{S}_+^n$ denotes the set of $n \times n$ positive semidefinite matrices, $\hat{\Sigma}$ is the sample covariance matrix, $\|S\|_1 = \sum_{ij} |S_{ij}|$ and $\rho > 0$ is a weighting parameter. Note that the first part of the objective function in (2.1), i.e., $\langle \hat{\Sigma}, S \rangle - \log\det(S)$, gives the maximum likelihood estimation of the inverse covariance matrix based on the sample covariance matrix $\hat{\Sigma}$. The ℓ_1 norm of S is added to the objective function to promote the sparsity of the solution.

Note that by introducing a variable T, Problem (2.1) can be rewritten as

$$\min_{S,T} \ f(S) + g(T), \quad \text{s.t.} \quad S - T = 0, \tag{2.2}$$

where $f(S) := \langle \hat{\Sigma}, S \rangle - \log\det(S)$ and $g(T) := \rho \|T\|_1$. Thus we have transformed (2.1) into the form of (1.1), i.e., convex minimization with two separable blocks. Two alternating direction methods have been suggested by Yuan [59] and Scheinberg, Ma and Goldfarb [47] to solve this reformulation (2.2). The one proposed by Yuan [59] is precisely the ADMM (1.6) applied to solving (2.2).

When ADMM (1.6) is applied to solve (2.2), the first subproblem in (1.6) can be reduced to:

$$S^{k+1} := \mathrm{argmin}_S \ \langle \hat{\Sigma}, S \rangle - \log\det(S) + \frac{1}{2\mu} \|S - T^k - \mu \Lambda^k\|_F^2, \tag{2.3}$$

where Λ^k denotes the Lagrange multiplier associated with the equality constraint $S - T = 0$ in the k-th iteration. Note that the solution of this problem corresponds to the proximal mapping of the $-\log\det(\cdot)$ function. The first-order optimality conditions of (2.3) are given by:

$$0 \in \hat{\Sigma} - S^{-1} + \frac{1}{\mu}(S - T^k - \mu \Lambda^k). \tag{2.4}$$

It is easy to verify that

$$S^{k+1} := U \mathrm{Diag}(\gamma) U^\top, \tag{2.5}$$

satisfies (2.4) and thus is the optimal solution of (2.3), where $U\mathrm{Diag}(\sigma)U^\top$ is the eigenvalue decomposition of $\mu\hat{\Sigma} - T^k - \mu\Lambda^k$ and $\gamma_i = (-\sigma_i + \sqrt{\sigma_i^2 + 4\mu})/2, i = 1, \ldots, n$. The second subproblem in (1.6) can be reduced

to:

$$T^{k+1} := \text{argmin}_T \ \rho\|T\|_1 + \frac{1}{2\mu}\|T - S^{k+1} + \mu\Lambda^k\|_F^2, \tag{2.6}$$

which has a closed-form solution that is given by the ℓ_1 shrinkage operation

$$T^{k+1} := \text{Shrink}(S^{k+1} - \mu\Lambda^k, \mu\rho), \tag{2.7}$$

where the ℓ_1 shrinkage operation is defined as

$$[\text{Shrink}(Z,\xi)]_{ij} := \begin{cases} Z_{ij} - \tau, & \text{if} \quad Z_{ij} > \tau \\ -Z_{ij} + \tau, & \text{if} \quad Z_{ij} < -\tau \\ 0, & \text{if} \quad -\tau \le Z_{ij} \le \tau. \end{cases} \tag{2.8}$$

Thus we have shown that the two subproblems of ADMM (1.6) for solving SICS (2.1) have closed-form solutions that are easy to obtain. Therefore each iteration of ADMM (1.6) corresponds to two easy subproblems, which makes ADMM appealing to SICS (2.1).

Another problem that is suitable for ADMM (1.6) is RPCA. RPCA seeks to decompose a given matrix $M \in \mathbb{R}^{m\times n}$ into two parts $M := L + S$ with L being a low-rank matrix and S being a sparse matrix. The following convex formulation of RPCA is proposed by Chandrasekaran et al. [10] and Candès et al. [5]:

$$\min_{L,S} \ \|L\|_* + \rho\|S\|_1, \quad \text{s.t.} \quad L + S = M, \tag{2.9}$$

where $\rho > 0$ is a weighting parameter. Notice that (2.9) is already in the form of (1.1) with both A and B being identity matrices, thus it can be solved by ADMM (1.6). When ADMM (1.6) is applied to solve (2.9), the first subproblem can be reduced to

$$L^{k+1} := \text{argmin}_L \ \|L\|_* + \frac{1}{2\mu}\|L + S^k - M - \mu\Lambda^k\|_F^2. \tag{2.10}$$

It is known that the solution of (2.10) is given by the matrix shrinkage operation which correponds to a singular value decomposition (SVD), (see, e.g., [4] and [39])

$$L^{k+1} := \text{MatShrink}(M - S^k + \mu\Lambda^k, \mu), \tag{2.11}$$

where the matrix shrinkage operator $\text{MatShrink}(Z,\xi)$ is defined as

$$\text{MatShrink}(Z,\xi) := U\text{Diag}(\max\{\sigma - \mu, 0\})V^\top, \tag{2.12}$$

and $U\text{Diag}(\sigma)V^\top$ is the SVD of matrix Z. The second subproblem in (1.6) can be reduced to

$$S^{k+1} := \text{argmin}_S \ \rho\|S\|_1 + \frac{1}{2\mu}\|L^{k+1} + S - M - \mu\Lambda^k\|_F^2. \tag{2.13}$$

This problem again has an easy closed-form solution, the ℓ_1 shrinkage operation:

$$S^{k+1} := \text{Shrink}(M - L^{k+1} + \mu\Lambda^k, \mu\rho). \tag{2.14}$$

Thus we have shown that the two subproblems in ADMM (1.6) for solving RPCA (2.9) are both easy, which makes ADMM (1.6) appealing to RPCA (2.9).

2.2. Problems with more than two blocks. In this section, we discuss some problems with three or more blocks of variable and objective function that arise from statistics, machine learning, image processing etc. For these problems, ADMM (1.6) does not apply directly.

Example 1. Latent Variable Gaussian Graphical Model Selection (LVGGMS). LVGGMS is an extension of SICS (2.1) when there exist latent variables. The recent paper by Chandrasekaran et al. [9] considers the scenario where the full data consist of both observed variables and hidden variables. Let $X_{n\times 1}$ be the observed variables. Suppose that there are some hidden variables $Y_{r\times 1}$ ($r \ll n$) such that (X, Y) jointly follow a multivariate normal distribution. Denote the covariance matrix by $\Sigma_{(X,Y)}$ and the precision matrix by $\Theta_{(X,Y)}$. Then we can write $\Sigma_{(X,Y)} = [\Sigma_X, \Sigma_{XY}; \Sigma_{YX}, \Sigma_Y]$ and $\Theta_{(X,Y)} = [\Theta_X, \Theta_{XY}; \Theta_{YX}, \Theta_Y]$. Given the hidden variables Y, the conditional concentration matrix of observed variables, Θ_X, is sparse for a sparse graphical model. However, the marginal concentration matrix of observed variables, $\Sigma_X^{-1} = \Theta_X - \Theta_{XY}\Theta_Y^{-1}\Theta_{YX}$, might not be a sparse matrix but a difference between the sparse term Θ_X and the low-rank term $\Theta_{XY}\Theta_Y^{-1}\Theta_{YX}$ (note that the rank of this $n \times n$ matrix is r). The problem of interest is to recover the sparse conditional matrix Θ_X based on observed variables X. [9] accomplishes this goal by solving a convex optimization problem under the assumption that $\Sigma_X^{-1} = S - L$ for some sparse matrix S and low-rank matrix L. The low rank assumption on L holds naturally since r is much less than n. Motivated by the success of the convex relaxation for rank-minimization problem, [9] introduced a regularized maximum normal likelihood decomposition framework called the latent variable graphical model selection as follows.

$$\min_{R,S,L} \langle R, \hat{\Sigma}_X\rangle - \log\det(R) + \alpha\|S\|_1 + \beta\mathrm{Tr}(L), \quad \text{s.t.} \quad R = S - L, R \succ 0, L \succeq 0, \tag{2.15}$$

where $\hat{\Sigma}_X$ is the sample covariance matrix of X and $\mathrm{Tr}(L)$ denotes the trace of matrix L. Note that the constraint $R \succ 0$ can be removed as the function $\log\det(R)$ in the objective function already implicitly imposes this constraint. The constraint $L \succeq 0$ can be put into the objective function by using an indicator function:

$$\mathcal{I}(L \succeq 0) := \begin{cases} 0, & \text{if } L \succeq 0 \\ +\infty, & \text{otherwise.} \end{cases} \tag{2.16}$$

This leads to the following equivalent reformulation of (2.15):

$$\min_{R,S,L} \langle R, \hat{\Sigma}_X\rangle - \log\det(R) + \alpha\|S\|_1 + \beta\mathrm{Tr}(L) + \mathcal{I}(L \succeq 0), \quad \text{s.t.} \quad R - S + L = 0. \tag{2.17}$$

Note that there are three blocks of variables R, S and L in (2.17). To reduce it to the form of (1.1) with two blocks of variables, one may group two blocks of variables (say S and L) as one block $T := [S; L]$. However, when ADMM (1.6) is applied to solve (2.17), the subproblem with respect to T is not easy, because the constraint becomes $R + [-I, I]T = 0$ and the coefficient matrix of T is not identity.

Example 2. Stable Principal Component Pursuit (SPCP) with Nonnegative Constraint. SPCP is actually the stable version of RPCA (2.9) when there are noises in the observation matrix M. The convex formulation

for SPCP is the following problem (see [61]):

$$\begin{array}{ll} \min_{L,S,Z} & \|L\|_* + \rho\|S\|_1 \\ \text{s.t.} & L+S+Z=M \\ & \|Z\|_F \le \sigma, \end{array} \tag{2.18}$$

where the matrix Z denotes the noise and $\sigma > 0$ is the noise level. In many applications of RPCA (2.9) and SPCP (2.18) such as background extraction from surveillance video, face recognition, video denoising etc. (see e.g., [5, 25]), the low-rank matrix usually stands for an image. Thus it is very natural to add a componentwise non-negative constraint $L \ge 0$ to (2.18) since the pixels of images take nonnegative values. This results in the following SPCP with nonnegative constraint:

$$\begin{array}{ll} \min_{L,S,Z} & \|L\|_* + \rho\|S\|_1 + \mathcal{I}(\|Z\|_F \le \sigma) + \mathcal{I}(L \ge 0) \\ \text{s.t.} & L+S+Z=M. \end{array} \tag{2.19}$$

Note that the proximal mapping of $\|L\|_* + \mathcal{I}(L \ge 0)$ is not easy to compute, we need to further split the two functions. To this purpose, we introduce a new variable K and rewrite (2.19) as

$$\begin{array}{ll} \min_{L,S,Z,K} & \|L\|_* + \rho\|S\|_1 + \mathcal{I}(\|Z\|_F \le \sigma) + \mathcal{I}(K \ge 0) \\ \text{s.t.} & L+S+Z=M \\ & L-K=0. \end{array} \tag{2.20}$$

To reduce (2.20) into the form of (1.1), one can group L and S as one big block $[L;S]$ and group Z and K as one big block $[Z;K]$. However, note that the constraint becomes

$$\begin{pmatrix} I & I \\ I & 0 \end{pmatrix} \begin{pmatrix} L \\ S \end{pmatrix} + \begin{pmatrix} I & 0 \\ 0 & -I \end{pmatrix} \begin{pmatrix} Z \\ K \end{pmatrix} = \begin{pmatrix} M \\ 0 \end{pmatrix}. \tag{2.21}$$

Then the subproblem with respect to $[L;S]$ is not easy to solve because the coefficient matrix of $[L;S]$ is not an identity matrix.

Example 3. Total variation based MRI deblurring. Magnetic Resonance Image (MRI) deblurring problem is an important topic in medical imaging. In MRI deblurring, one tries to reconstruct the image x based on some observation b, which is usually obtained through a partial Fourier transform R. The work on sparse MRI [37] shows that one can recover the image using only a limited number of measurements b. Of course, we have some a prior information on the image. MRI is usually believed to possess a piece-wise constant behavior, which ensures a small total variation term. Also, it is believed that MRI is usually sparse under wavelet transform. Thus the MRI deblurring problem can be formulated as

$$\min_x \ \mathrm{TV}(x) + \alpha\|Wx\|_1, \quad \text{s.t.} \quad \|Rx-b\|_2 \le \sigma, \tag{2.22}$$

where $\mathrm{TV}(x) := \sum_{i=1}^N \|D_i x\|_2$ denotes the total variation function, $D_i x \in \mathbb{R}^2$ represents certain first-order finite differences of x at pixel i in horizontal and vertical directions, W is a wavelet transform and $\sigma > 0$ is the noise level. Note that most algorithms in the literature for solving MRI deblurring problem with both

TV and ℓ_1-regularization terms solve the unconstrained problem

$$\min_x \ \mathrm{TV}(x) + \alpha\|Wx\|_1 + \frac{\beta}{2}\|Rx - b\|_2. \tag{2.23}$$

For example, see [40, 52, 57]. We prefer (2.22) because the noise level σ is usually easier to estimate than the weighting parameter β.

To ensure that each nonsmooth convex function involved in the objective function has an easy proximal mapping, we introduce new variables y and $z_i, i = 1, \ldots, N$ and rewrite Problem (2.23) as

$$\begin{array}{ll} \min_x & \sum_i \|z_i\|_2 + \alpha\|Wx\|_1 + \mathcal{I}(\|y\|_2 \le \sigma) \\ \text{s.t.} & z_i = D_i x, \forall i = 1, \ldots, N \\ & y = Rx - b. \end{array} \tag{2.24}$$

To reduce (2.24) into the form of (1.1), one can group $[y; z_1; \ldots; z_N]$ as one big block. Now since the constraint becomes

$$\begin{pmatrix} R \\ D_1 \\ \vdots \\ D_N \end{pmatrix} x - \begin{pmatrix} y \\ z_1 \\ \vdots \\ z_N \end{pmatrix} = \begin{pmatrix} b \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \tag{2.25}$$

and the coefficient matrix of x is not identity matrix, the subproblem with respect to x is not easy to solve when ADMM (1.6) is applied to solve it.

Example 4. Compressive Principal Component Pursuit (CPCP). CPCP proposed by Wright et al. in [54] is a generalization of the RPCA (2.9). CPCP also seeks a low-rank and sparse decomposition of a matrix $\hat{M}$. However, the matrix $\hat{M}$ is not available directly, while it is observed through a linear transformation $\mathcal{A}$. This results in the following convex formulation of CPCP (see [54]),

$$\min \ \|L\|_* + \rho\|S\|_1, \ \text{s.t.} \ \ \mathcal{A}(L + S) = M. \tag{2.26}$$

In particular, Wright et al. [54] considered the case when $\mathcal{A}$ represents a projection operator onto a subspace. In this case, M represents the projection of matrix $\hat{M}$ on a subspace.

CPCP (2.26) already has two blocks. However, when ADMM (1.6) is applied to solve it, the two subproblems are not easy, because the coefficient matrices of L and S are both $\mathcal{A}$, which is not an identity matrix.

Example 5. Robust Alignment for Linearly Correlated Images. The problem of robust alignment for linearly correlated images is modeled as a sparse and low-rank decomposition problem by Peng et al. in [44]. A set of well-aligned images of the same object is usually believed to be linearly correlated. Thus the matrix formed by stacking the images should be a low-rank matrix. However, misalignment usually breaks the linear structure in the data and leads to a full-rank matrix. Also, there sometimes exist sparse errors in the images due to corruption, occlusion, shadows and specularities etc. Peng et al. [44] proposed the following optimization problem to model this robust alignment problem:

$$\min_{L,S,x} \ \|L\|_* + \rho\|S\|_1, \quad \text{s.t.} \quad D \circ x = L + S, \tag{2.27}$$

where D denotes the matrix formed by the given images and x represents the alignment operator. Thus $D \circ x$ denotes the matrix formed by the aligned images and it is expected to be decomposed into a low-rank matrix L plus a sparse matrix S. Notice that Problem (2.27) is not a convex problem because the operation $D \circ x$ is nonlinear. Peng et al. [44] proposed to solve (2.27) by a sequence of convex optimization problems (see [44] for the details of the derivation). In the k-th iteration, one needs to solve a linearized problem of (2.27) at x^k:

$$\min_{L,S,\Delta x} \ \|L\|_* + \rho\|S\|_1, \quad \text{s.t.} \quad D \circ x^k + \hat{D}\Delta x = L + S, \tag{2.28}$$

where $D \circ x^k$ is a constant term and Δx is the step for updating x^k. Note that there are three variables $L, S, \Delta x$ in (2.28). To reduce it into the form of (1.1), one can group L and S as one big variable $[L; S]$. However, notice that the constraint becomes

$$\begin{pmatrix} I & I \end{pmatrix} \begin{pmatrix} L \\ S \end{pmatrix} - \hat{D}\Delta x = D \circ x^k,$$

the subproblems of ADMM (1.6) are not easy because the coefficient matrices of $[L; S]$ and Δx are not identity matrices.

3. Alternating proximal gradient method. In this section, we propose an alternating proximal gradient method (APGM) for solving the convex optimization problem (1.1). APGM is based on the framework of ADMMs (1.6) and (1.7), but all the subproblems in APGM are easy to solve. In fact, when APGM is applied to solve (1.3), the N subproblems of APGM correspond to the proximal mappings of $f_1, \ldots, f_N$, respectively. Thus all the problems that are difficult to solve by ADMM (1.6) as we discussed in Section 2.2, can be solved easily by APGM.

Note that ADMM (1.6) can be written explicitly as

$$\begin{cases} x^{k+1} & := & \operatorname{argmin}_x \ f(x) + g(y^k) + \frac{1}{2\mu}\|Ax + By^k - b - \mu\lambda^k\|_2^2 \\ y^{k+1} & := & \operatorname{argmin}_y \ f(x^{k+1}) + g(y) + \frac{1}{2\mu}\|Ax^{k+1} + By - b - \mu\lambda^k\|_2^2 \\ \lambda^{k+1} & := & \lambda^k - (Ax^{k+1} + By^{k+1} - b)/\mu. \end{cases} \tag{3.1}$$

The subproblems in ADMM (3.1) are not easy to solve because the matrices A and B are involved in the quadratic penalty term of the augmented Lagrangian function. An iterative solver is usually needed to solve the subproblems. However, it is costly and not necessary to solve the subproblems exactly when they are hard to solve. Therefore, we propose to solve the subproblems inexactly by taking one proximal gradient step. This results in the following algorithm (3.2), which we call alternating proximal gradient method because we alternatingly perform a proximal gradient step for the two subproblems.

$$\begin{cases} x^{k+1} & := & \operatorname{argmin}_x \ f(x) + \frac{1}{2\mu\tau_1}\|x - (x^k - \tau_1 A^\top(Ax^k + By^k - b - \mu\lambda^k))\|_2^2 \\ y^{k+1} & := & \operatorname{argmin}_y \ g(y) + \frac{1}{2\mu\tau_2}\|y - (y^k - \tau_2 B^\top(Ax^{k+1} + By^k - b - \mu\lambda^k)\|_2^2 \\ \lambda^{k+1} & := & \lambda^k - (Ax^{k+1} + By^{k+1} - b)/\mu, \end{cases} \tag{3.2}$$

where τ_1 and τ_2 are the step sizes for the proximal gradient steps. We need to make a few remarks about APGM (3.2). We use the first subproblem as an example and the arguments hold similarly for the second subproblem. For the subproblem with respect to x, the term $A^\top(Ax^k + By^k - b - \mu\lambda^k)$ is actually the gradient of the quadratic penalty term $\frac{1}{2}\|Ax + By^k - b - \mu\lambda^k\|_2^2$. The quadratic term $\frac{1}{2}\|x - (x^k - \tau_1 A^\top(Ax^k + By^k - b -$

$\mu\lambda^k))\|_2^2$ in the first subproblem of (3.2) is thus a quadratic proximal term with the proximal point obtained by taking a gradient step at the current iterate x^k. Note that since

$$\frac{1}{2\tau_1}\|x-(x^k-\tau_1 A^\top(Ax^k+By^k-b-\mu\lambda^k))\|_2^2 = \langle x-x^k, A^\top(Ax^k+By^k-b-\mu\lambda^k)\rangle + \frac{1}{2\tau_1}\|x-x^k\|_2^2 + \text{constant},$$

the proximal gradient step can also be seen as linearizing the quadratic penalty term in the augmented Lagrangian function using the gradient at x^k plus a proximal term $\frac{1}{2\tau_1}\|x-x^k\|_2^2$ at x^k.

The idea of incorporating proximal step into the alternating direction method of multipliers is actually not new. It has been suggested by Chen and Teboulle [11] and Eckstein [19]. This idea has then been generalized by He et al. [31] to allow varying penalty and proximal parameters. Recently, this technique has been used for sparse and low-rank optimization problems (see Yang and Zhang [56] and Tao and Yuan [49]).

We will show in the following that although we do not solve the subproblems in ADMM (1.6) exactly but use only one proximal gradient step to approximately solve them in APGM (3.2), the APGM algorithm (3.2) converges to an optimal solution of (1.1) from any starting point, as long as the step sizes τ_1 and τ_2 are bounded by $\tau_1 < 1/\lambda_{\max}(A^\top A)$ and $\tau_2 < 1/\lambda_{\max}(B^\top B)$. During the preparation of this paper, we noticed that Deng and Yin [15] and Luo [36] also studied the convergence property of alternating direction methods with proximal gradient steps, but the problems and algorithms were motivated in different ways.

3.1. Global convergence result. In this section, we show the global convergence result of Algorithm (3.2) for solving (1.1). We need to prove the following lemma first.

LEMMA 3.1. *Assume that (x^*, y^*) is an optimal solution of (1.1) and λ^* is the corresponding optimal dual variable associated with the equality constraint $Ax + By = b$. Assume the step sizes τ_1 and τ_2 of the proximal gradient steps satisfy $\tau_1 < 1/\lambda_{\max}(A^\top A)$ and $\tau_2 < 1/\lambda_{\max}(B^\top B)$, where $\lambda_{\max}(C)$ denotes the largest eigenvalue of matrix C. Then there exists $\eta > 0$ such that the sequence (x^k, y^k, λ^k) produced by (3.2) satisfies*

$$\|u^k - u^*\|_H^2 - \|u^{k+1} - u^*\|_H^2 \geq \eta\|u^k - u^{k+1}\|_H^2, \tag{3.3}$$

where $u^* = \begin{pmatrix} x^* \\ y^* \\ \lambda^* \end{pmatrix}$, $u^k = \begin{pmatrix} x^k \\ y^k \\ \lambda^k \end{pmatrix}$ *and* $H = \begin{pmatrix} \frac{1}{\mu\tau_1}I - \frac{1}{\mu}A^\top A & 0 & 0 \\ 0 & \frac{1}{\mu\tau_2}I & \\ 0 & 0 & \mu I \end{pmatrix}$. *Note that $\tau_1 < 1/\lambda_{\max}(A^\top A)$ guarantees that H is positive definite. The norm $\|\cdot\|_H^2$ is defined as $\|u\|_H^2 = \langle u, Hu\rangle$ and the corresponding inner product $\langle\cdot,\cdot\rangle_H$ is defined as $\langle u, v\rangle_H = \langle u, Hv\rangle$.*

Proof. Since (x^*, y^*, λ^*) is optimal to (1.1), it follows from the KKT conditions that the followings hold:

$$0 \in \partial f(x^*) - A^\top\lambda^*, \tag{3.4}$$

$$0 \in \partial g(y^*) - B^\top\lambda^*, \tag{3.5}$$

and

$$0 = Ax^* + By^* - b. \tag{3.6}$$

Note that the optimality conditions for the first subproblem (i.e., the subproblem with respect to x) in

(3.2) are given by

$$0 \in \tau_1\mu\partial f(x^{k+1}) + x^{k+1} - x^k + \tau_1 A^\top(Ax^k + By^k - b - \mu\lambda^k). \tag{3.7}$$

By using the updating formula for λ^k, i.e.,

$$\lambda^{k+1} = \lambda^k - (Ax^{k+1} + By^{k+1} - b)/\mu, \tag{3.8}$$

(3.7) can be reduced to

$$0 \in \tau_1\mu\partial f(x^{k+1}) + x^{k+1} - x^k + \tau_1 A^\top(Ax^k - Ax^{k+1} + By^k - By^{k+1} - \mu\lambda^{k+1}). \tag{3.9}$$

Combining (3.4) and (3.9) and using the fact that $\partial f(\cdot)$ is a monotone operator, we get

$$(x^{k+1} - x^*)^\top \left(\frac{1}{\tau_1\mu}(x^k - x^{k+1}) - \frac{1}{\mu}A^\top A(x^k - x^{k+1}) - \frac{1}{\mu}A^\top B(y^k - y^{k+1}) + A^\top(\lambda^{k+1} - \lambda^*)\right) \geq 0. \tag{3.10}$$

The optimality conditions for the second subproblem (i.e., the subproblem with respect to y) in (3.2) are given by

$$0 \in \tau_2\mu\partial g(y^{k+1}) + y^{k+1} - y^k + \tau_2 B^\top(Ax^{k+1} + By^k - b - \mu\lambda^k). \tag{3.11}$$

Using (3.8), (3.11) can be reduced to

$$0 \in \tau_2\mu\partial g(y^{k+1}) + y^{k+1} - y^k + \tau_2 B^\top(By^k - By^{k+1} - \mu\lambda^{k+1}). \tag{3.12}$$

Combining (3.5) and (3.12) and using the fact that $\partial g(\cdot)$ is a monotone operator, we get

$$(y^{k+1} - y^*)^\top \left(\frac{1}{\tau_2\mu}(y^k - y^{k+1}) - \frac{1}{\mu}B^\top B(y^k - y^{k+1}) + B^\top(\lambda^{k+1} - \lambda^*)\right) \geq 0. \tag{3.13}$$

Summing (3.10) and (3.13), and using $Ax^* + By^* = b$, we obtain

$$\begin{array}{l} \frac{1}{\tau_1\mu}(x^{k+1} - x^*)^\top(x^k - x^{k+1}) - \frac{1}{\mu}(x^{k+1} - x^*)^\top A^\top A(x^k - x^{k+1}) + \frac{1}{\tau_2\mu}(y^{k+1} - y^*)^\top(y^k - y^{k+1}) \\ -(\lambda^k - \lambda^{k+1})^\top B(y^k - y^{k+1}) + \mu(\lambda^k - \lambda^{k+1})^\top(\lambda^{k+1} - \lambda^*) \geq 0. \end{array} \tag{3.14}$$

Using the notation of u^k, u^* and H, (3.14) can be rewritten as

$$\langle u^{k+1} - u^*, u^k - u^{k+1}\rangle_H \geq \langle \lambda^k - \lambda^{k+1}, By^k - By^{k+1}\rangle, \tag{3.15}$$

which can be further written as

$$\langle u^k - u^*, u^k - u^{k+1}\rangle_H \geq \|u^k - u^{k+1}\|_H^2 + \langle \lambda^k - \lambda^{k+1}, By^k - By^{k+1}\rangle. \tag{3.16}$$

Combining (3.16) with the identity

$$\|u^{k+1} - u^*\|_H^2 = \|u^{k+1} - u^k\|_H^2 - 2\langle u^k - u^{k+1}, u^k - u^*\rangle_H + \|u^k - u^*\|_H^2,$$

we get

$$\begin{aligned} & \|u^k - u^*\|_H^2 - \|u^{k+1} - u^*\|_H^2 \\ = \; & 2\langle u^k - u^{k+1}, u^k - u^*\rangle_H - \|u^{k+1} - u^k\|_H^2 \\ \geq \; & \|u^{k+1} - u^k\|_H^2 + 2\langle \lambda^k - \lambda^{k+1}, By^k - By^{k+1}\rangle. \end{aligned} \tag{3.17}$$

Let $\xi := \frac{1}{2} + \frac{\tau_2 \lambda_{\max}(B^\top B)}{2}$, then we know that $\tau_2 \lambda_{\max}(B^\top B) < \xi < 1$ since $\tau_2 < \frac{1}{\lambda_{\max}(B^\top B)}$. Let $\rho := \mu\xi$. Then from the Cauchy-Schwartz inequality we have

$$\begin{aligned} 2\langle \lambda^k - \lambda^{k+1}, By^k - By^{k+1}\rangle \; & \geq \; -\rho\|\lambda^k - \lambda^{k+1}\|^2 - \tfrac{1}{\rho}\|By^k - By^{k+1}\|^2 \\ & \geq \; -\rho\|\lambda^k - \lambda^{k+1}\|^2 - \tfrac{1}{\rho}\lambda_{\max}(B^\top B)\|y^k - y^{k+1}\|^2. \end{aligned} \tag{3.18}$$

Combining (3.17) and (3.18) we get

(3.19)

$$\begin{aligned} & \|u^k - u^*\|_H^2 - \|u^{k+1} - u^*\|_H^2 \\ \geq \; & (x^k - x^{k+1})^\top \left(\tfrac{1}{\mu\tau_1} I - \tfrac{1}{\mu} A^\top A\right)(x^k - x^{k+1}) + \left(\tfrac{1}{\mu\tau_2} - \tfrac{1}{\rho}\lambda_{\max}(B^\top B)\right)\|y^k - y^{k+1}\|^2 + (\mu - \rho)\|\lambda^k - \lambda^{k+1}\|^2 \\ \geq \; & \eta\|u^k - u^{k+1}\|_H^2, \end{aligned}$$

where $\eta := \min\{\frac{1}{\mu\tau_1} - \frac{1}{\mu}\lambda_{\max}(A^\top A), \frac{1}{\mu\tau_2} - \frac{1}{\rho}\lambda_{\max}(B^\top B), \mu - \rho\} > 0$. This completes the proof. □

We now give the global convergence result of Algorithm (3.2) for solving (1.1).

THEOREM 3.2. *The sequence $\{(x^k, y^k, \lambda^k)\}$ produced by Algorithm* (3.2) *with $\tau_1 < 1/\lambda_{\max}(A^\top A)$ and $\tau_2 < 1/\lambda_{\max}(B^\top B)$ converges to an optimal solution to Problem* (1.1) *from any starting point.*

Proof. From Lemma 3.1 we can easily get that

- (i) $\|u^k - u^{k+1}\|_H \to 0$;
- (ii) $\{u^k\}$ lies in a compact region;
- (iii) $\|u^k - u^*\|_H^2$ is monotonically non-increasing and thus converges.

It follows from (i) that $x^k - x^{k+1} \to 0$, $y^k - y^{k+1} \to 0$ and $\lambda^k - \lambda^{k+1} \to 0$. Then (3.8) implies that $Ax^k + By^k - b \to 0$. From (ii) we obtain that, u^k has a subsequence $\{u^{k_j}\}$ that converges to $\hat{u} = (\hat{x}, \hat{y}, \hat{\lambda})$. Therefore, $(\hat{x}, \hat{y}, \hat{\lambda})$ is a limit point of $\{(x^k, y^k, \lambda^k)\}$ and $A\hat{x} + B\hat{y} - b = 0$.

Note that (3.9) implies that

$$0 \in \partial f(\hat{x}) - A^\top \hat{\lambda}. \tag{3.20}$$

Note also that (3.12) implies that

$$0 \in \partial g(\hat{y}) - B^\top \hat{\lambda}. \tag{3.21}$$

(3.20), (3.21) and $A\hat{x} + B\hat{y} - b = 0$ imply that $(\hat{x}, \hat{y}, \hat{\lambda})$ satisfies the KKT conditions for (1.1) and thus is an optimal solution to (1.1). Therefore, we showed that any limit point of $\{(x^k, y^k, \lambda^k)\}$ is an optimal solution to (1.1).

To complete the proof, it remains to show that $\{(x^k, y^k, \lambda^k)\}$ has a unique limit point. Let $\{(\hat{x}_1, \hat{y}_1, \hat{\lambda}_1)\}$ and $\{(\hat{x}_2, \hat{y}_2, \hat{\lambda}_2)\}$ be any two limit points of $\{(x^k, y^k, \lambda^k)\}$. As we have shown, both $\{(\hat{x}_1, \hat{y}_1, \hat{\lambda}_1)\}$ and $\{(\hat{x}_2, \hat{y}_2, \hat{\lambda}_2)\}$ are optimal solutions to (1.1). Thus, u^* in (3.19) can be replaced by $\hat{u}_1 := (\hat{x}_1, \hat{y}_1, \hat{\lambda}_1)$ and

$\hat{u}_2 := (\hat{x}_2, \hat{y}_2, \hat{\lambda}_2)$. This results in

$$\|u^{k+1} - \hat{u}_i\|_H^2 \leq \|u^k - \hat{u}_i\|_H^2, \quad i = 1, 2,$$

and we thus get the existence of the limits

$$\lim_{k\to\infty} \|u^k - \hat{u}_i\|_H = \eta_i < +\infty, \quad i = 1, 2.$$

Now using the identity

$$\|u^k - \hat{u}_1\|_H^2 - \|u^k - \hat{u}_2\|_H^2 = -2\langle u^k, \hat{u}_1 - \hat{u}_2\rangle_H + \|\hat{u}_1\|_H^2 - \|\hat{u}_2\|_H^2$$

and passing the limit we get

$$\eta_1^2 - \eta_2^2 = -2\langle \hat{u}_1, \hat{u}_1 - \hat{u}_2\rangle_H + \|\hat{u}_1\|_H^2 - \|\hat{u}_2\|_H^2 = -\|\hat{u}_1 - \hat{u}_2\|_H^2$$

and

$$\eta_1^2 - \eta_2^2 = -2\langle \hat{u}_2, \hat{u}_1 - \hat{u}_2\rangle_H + \|\hat{u}_1\|_H^2 - \|\hat{u}_2\|_H^2 = \|\hat{u}_1 - \hat{u}_2\|_H^2.$$

Thus we must have $\|\hat{u}_1 - \hat{u}_2\|_H^2 = 0$ and hence the limit point of $\{(x^k, y^k, \lambda^k)\}$ is unique. □

3.2. Relation to existing works. In this section, we discuss the relationship of our proposed APGM with the predictor corrector proximal multiplier method (PCPM) studied by Chen and Teboulle [11], the linearized augmented Lagrangian method (LAL) studied by Yang and Yuan [55] and the Bregmanized operator splitting (BOS) method studied by Zhang et al. [60].

In [11], Chen and Teboulle proposed a PCPM method to solve the following convex problem

$$\min_{x,y} \ f(x) + g(y), \quad \text{s.t.} \quad Ax - y = 0. \tag{3.22}$$

A typical iteration of PCPM is

$$\begin{cases} \bar{\lambda}^{k+1} & := & \lambda^k - \mu_k(Ax^k - y^k) \\ x^{k+1} & := & \text{argmin}_x \ f(x) - \langle \bar{\lambda}^{k+1}, Ax\rangle + \frac{1}{2\mu_k}\|x - x^k\|^2 \\ y^{k+1} & := & \text{argmin}_y \ g(y) + \langle \bar{\lambda}^{k+1}, y\rangle + \frac{1}{2\mu_k}\|y - y^k\|^2 \\ \lambda^{k+1} & := & \lambda^k - \mu_k(Ax^{k+1} - y^{k+1}), \end{cases} \tag{3.23}$$

where $\{\mu_k\}$ is a sequence of positive scalars. The first step and the last step of (3.23) can be seen as a predictor step and a corrector step to the Lagrange multiplier, respectively. A typical iteration of our APGM for solving (3.22) is

$$\begin{cases} x^{k+1} & := & \text{argmin}_x \ f(x) + \frac{1}{2\mu\tau_1}\|x - (x^k - \tau_1 A^\top(Ax^k - y^k - \mu\lambda^k))\|^2 \\ y^{k+1} & := & \text{argmin}_y \ g(y) + \frac{1}{2\mu\tau_2}\|y - (y^k + \tau_2(Ax^{k+1} - y^k - \mu\lambda^k)\|^2 \\ \lambda^{k+1} & := & \lambda^k - (Ax^{k+1} - y^{k+1})/\mu, \end{cases} \tag{3.24}$$

which can be rewritten as

$$
\left\{\begin{array}{lcl}
\bar{\lambda}^{k+1} & := & \lambda^k - (Ax^k - y^k)/\mu \\
x^{k+1} & := & \mathrm{argmin}_x \ f(x) - \langle \bar{\lambda}^{k+1}, Ax\rangle + \frac{1}{2\mu\tau_1}\|x - x^k\|^2 \\
\tilde{\lambda}^{k+1} & := & \lambda^k - (Ax^{k+1} - y^k)/\mu \\
y^{k+1} & := & \mathrm{argmin}_y \ g(y) + \langle \tilde{\lambda}^{k+1}, y\rangle + \frac{1}{2\mu\tau_2}\|y - y^k\|^2 \\
\lambda^{k+1} & := & \lambda^k - (Ax^{k+1} - y^{k+1})/\mu,
\end{array}\right. \tag{3.25}
$$

Thus, APGM can also be seen as a predictor corrector proximal multiplier method. However, the difference between APGM and PCPM is obvious. First, PCPM uses the same predicted multiplier for both subproblems for x and y, while APGM uses different predicted multipliers for the two subproblems. The predictor $\tilde{\lambda}^{k+1}$ used for the second subproblem uses the latest information x^{k+1} obtained from the first subproblem. Second, the penalty parameters used in the proximal terms of PCPM and APGM are quite different. Third, APGM handles more general problem (1.1) (note that (3.22) is a special case of (1.1) when $B = -I$).

Other methods that are closely related to APGM are the LAL method proposed by Yang and Yuan [55] and the BOS method proposed by Zhang et al. [60]. In fact, as discussed in [55], LAL and BOS are the same method but derived from different motivations. Yang and Yuan [55] proposed LAL method for solving nuclear norm minimization problems. Zhang et al. [60] proposed BOS method for solving nonlocal TV denoising problems. Although the LAL method proposed in [55] was for solving the nuclear norm minimization problem

$$
\min_X \quad \|X\|_*, \quad \text{s.t.} \quad \mathcal{A}X = b,
$$

it can be easily adopted for solving (1.1) as follows:

$$
\left\{\begin{array}{lcl}
(x^{k+1}, y^{k+1}) & := & \mathrm{argmin}_{x,y} \ f(x) + g(y) + \frac{1}{2\mu\tau}\left\|\begin{pmatrix} x \\ y \end{pmatrix} - \left(\begin{pmatrix} x^k \\ y^k \end{pmatrix} - \tau \begin{pmatrix} A^\top \\ B^\top \end{pmatrix}\left(Ax^k + By^k - b - \mu\lambda^k\right)\right)\right\|_2^2 \\
\lambda^{k+1} & := & \lambda^k - (Ax^{k+1} + By^{k+1} - b)/\mu.
\end{array}\right. \tag{3.26}
$$

It is easy to see that the LAL method linearizes the quadratic penalty term in the augmented Lagrangian method (1.4). After this linearization, minimizing the augmented Lagrangian function with respect to x and y simultaneously is doable since x and y are now separable in the subproblem of (3.26). In fact, the subproblem of (3.26) can be reduced to the following two subproblems that correspond to the proximal mappings of f and g, respectively:

$$
x^{k+1} := \mathrm{argmin}_x \ f(x) + \frac{1}{2\mu\tau}\|x - (x^k - \tau A^\top(Ax^k + By^k - b))\|_2^2, \tag{3.27}
$$

and

$$
y^{k+1} := \mathrm{argmin}_y \ g(y) + \frac{1}{2\mu\tau}\|y - (y^k - \tau A^\top(Ax^k + By^k - b))\|_2^2. \tag{3.28}
$$

Now the relationship of the LAL method and APGM is obvious: the LAL method can be seen as a Jacobi type method in alternatingly linearizing and minimizing the augmented Lagrangian function, while the APGM can be seen as a Gauss-Seidel type method because the latest information x^{k+1} is used in linearizing and minimizing the augmented Lagrangian function for variable y. We argue that the effects of this difference, i.e.,

Jocobi type method vs. Gauss-Seidel type method, are significant. First, Gauss-Seidel method is believed to be faster than Jacobi type method because the latest information is always used. Second, APGM allows larger step sizes in the proximal gradient step than LAL method. Note that to guarantee global convergence, the step size τ in LAL is required to be bounded by $\tau < 1/\lambda_{\max}(E)$, where

$$E := \begin{pmatrix} A^\top \\ B^\top \end{pmatrix} \begin{pmatrix} A & B \end{pmatrix} = \begin{pmatrix} A^\top A & A^\top B \\ B^\top A & B^\top B \end{pmatrix},$$

(see [55]). However, Theorem 3.2 shows that the requirement for global convergence of APGM is $\tau_1 < 1/\lambda_{\max}(A^\top A)$ and $\tau_2 < 1/\lambda_{\max}(B^\top B)$. The largest eigenvalue of E is usually much larger than the largest eigenvalues of $A^\top A$ and $B^\top B$. For example, when A and B are both identity matrices, the largest eigenvalue of E is 2, which is twice larger than the eigenvalues of $A^\top A$ and $B^\top B$. Therefore, APGM allows larger step sizes in the proximal gradient steps. As we will show in the numerical experiments in Section 5, this significantly affects the convergence speed of these two methods. APGM converges much faster than LAL because the former allows larger step sizes. Third, the two subproblems (3.27) and (3.28) could be done in parallel because it is a Jacobi type method. However, we observe from our numerical experiments in Section 5 that the effect of this is not as significant as the ones caused by different (larger) step sizes.

3.3. How to group the variables. Apparently, using APGM (3.2) to solve the general problem (1.3) with more than two blocks requires to group the N variables into two blocks. Theoretically, we can group the N variables to two blocks with any combination. However, the numerical performance of APGM with different partition of the variables can be very different, because the step sizes of the proximal gradient steps are related to the largest singular values of A and B. Not surprisingly, we observe from numerical experiments that larger step sizes results in a faster algorithm. Thus, our criterion to group the variables in (1.3) is to ensure that the largest singular values of A and B are small so that both subproblems can take large step sizes in the proximal gradient steps.

We use Example 2 in Section 2.2, i.e., SPCP with nonnegative constraint (2.19), to illustrate how to group the variables into two blocks. Note that in (2.21), we group L and S as one big block and Z and K as one big block. Another way to group the variables into two blocks is to group L and K as one big block and S and Z as one big block, and the constraints can now be written as

$$\begin{pmatrix} I & 0 \\ I & -I \end{pmatrix} \begin{pmatrix} L \\ K \end{pmatrix} + \begin{pmatrix} I & I \\ 0 & 0 \end{pmatrix} \begin{pmatrix} S \\ Z \end{pmatrix} = \begin{pmatrix} M \\ 0 \end{pmatrix}. \tag{3.29}$$

We argue that (2.21) is a better partition for APGM (3.2) than (3.29). The coefficient matrix of $[L; S]$ in (2.21) and the coefficient matrix of $[L; K]$ in (3.29) have the same largest singular value $\sqrt{2.618}$. The largest singular value of the coefficient matrix of $[Z; K]$ in (2.21) is 1, while the largest singular value of the coefficient matrix of $[S; Z]$ in (3.29) is $\sqrt{2}$. Thus, the partition in (2.21) allows larger step size for the second subproblem. Therefore, when APGM (3.2) is applied to solve (2.19), it is better to use partition (2.21).

4. Applications. In this section, we show how to solve the problems with more than two blocks discussed in Section 2.2 using APGM (3.2).

Application 1. APGM for LVGGMS (2.17). There are three blocks of variables in LVGGMS (2.17). We should group two blocks of variables as one big block in order to use APGM (3.2). Without loss of generality, we choose to group S and L as one big block $W := [S; L]$. Then APGM for solving (2.17) can be

summarized as follows.

$$
(4.1)\qquad \left\{\begin{array}{lcl} R^{k+1} &:=& \mathrm{argmin}_R\ \langle R, \hat{\Sigma}\rangle - \log\det R + \frac{1}{2\mu\tau_1}\|R - (R^k - \tau_1(R^k - S^k + L^k - \mu\Lambda^k))\|_F^2 \\ S^{k+1} &:=& \mathrm{argmin}_S\ \alpha\|S\|_1 + \frac{1}{2\mu\tau_2}\|S - (S^k + \tau(R^{k+1} - S^k + L^k - \mu\Lambda^k))\|_F^2 \\ L^{k+1} &:=& \mathrm{argmin}_L\ \beta\mathrm{Tr}(L) + \mathcal{I}(L \succeq 0) + \frac{1}{2\mu\tau_2}\|L - (L^k - \tau(R^{k+1} - S^k + L^k - \mu\Lambda^k))\|_F^2 \\ \Lambda^{k+1} &:=& \Lambda^k - (R^{k+1} - S^{k+1} + L^{k+1})/\mu. \end{array}\right.
$$

We now show that the three subproblems in (4.1) can be easily solved. Note that the solution of the first subproblem in (4.1) corresponds to the proximal mapping of the $-\log\det(\cdot)$ function. The first-order optimality conditions of the first subproblem in (4.1) are given by:

$$
(4.2)\qquad 0 = \hat{\Sigma} - R^{-1} + (R^{k+1} - S^k + L^k - \mu\Lambda^k)/\mu.
$$

It is easy to verify that $R^{k+1} := U\mathrm{Diag}(\gamma)U^\top$ satisfies (4.2) and thus is optimal to the first subproblem of (4.1), where $U\mathrm{Diag}(\sigma)U^\top$ is the eigenvalue decomposition of matrix $\mu\tau_1\hat{\Sigma} - (R^k - \tau_1(R^k - S^k + L^k - \mu\Lambda^k))$, and $\gamma_i = (-\sigma_i + \sqrt{\sigma_i^2 + 4\mu\tau_1})/2, i = 1, \ldots, n$. Thus solving the first subproblem of (4.1) corresponds to an eigenvalue decomposition of an $n \times n$ matrix. The solution of the second subproblem of (4.1) is given by the ℓ_1 shrinkage operation

$$
S^{k+1} := \mathrm{Shrink}(S^k + \tau(R^{k+1} - S^k + L^k - \mu\Lambda^k), \alpha\mu\tau),
$$

where the shrinkage operator is defined in (2.8). The solution of the third subproblem of (4.1) corresponds to the proximal mapping of the indicator function $\mathcal{I}(L \succeq 0)$, which is equal to the projection of the matrix $W^k := L^k - \tau(R^{k+1} - S^k + L^k - \mu\Lambda^k) - \beta\mu\tau I$ onto the positive semidefinite cone. Thus the solution of this subproblem is given

$$
L^{k+1} := U\mathrm{Diag}(\max\{\sigma, 0\})U^\top,
$$

where $U\mathrm{Diag}(\sigma)U^\top$ is the eigenvalue decomposition of matrix W^k.

Application 2. APGM for SPCP with nonnegative constraints (2.20). There are four blocks of variables in the problem of SPCP with nonnegative constraints (2.20). We choose the way in (2.21) to partition the variables into two blocks. Then APGM for solving (2.20) can be described as

$$
(4.3)\qquad \left\{\begin{array}{lcl} L^{k+1} &:=& \mathrm{argmin}_L\ \|L\|_* + \frac{1}{2\mu\tau_1}\|L - (L^k - \tau_1(2L^k + S^k + Z^k - M - K^k - \mu\Lambda_1^k - \mu\Lambda_2^k))\|_F^2 \\ S^{k+1} &:=& \mathrm{argmin}_S\ \rho\|S\|_1 + \frac{1}{2\mu\tau_1}\|S - (S^k - \tau_1(L^k + S^k + Z^k - M - \mu\Lambda_1^k))\|_F^2 \\ Z^{k+1} &:=& \mathrm{argmin}_Z\ \mathcal{I}(\|Z\|_F \le \sigma) + \frac{1}{2\mu\tau_2}\|Z - (Z^k - \tau_2(L^{k+1} + S^{k+1} + Z^k - M - \mu\Lambda_1^k))\|_F^2 \\ K^{k+1} &:=& \mathrm{argmin}_K\ \mathcal{I}(K \ge 0) + \frac{1}{2\mu\tau_2}\|K - (K^k + \tau_2(L^{k+1} - K^k - \mu\Lambda_2^k))\|_F^2 \\ \Lambda_1^{k+1} &:=& \Lambda_1^k - (L^{k+1} + S^{k+1} + Z^{k+1} - M)/\mu \\ \Lambda_2^{k+1} &:=& \Lambda_2^k - (L^{k+1} - K^{k+1})/\mu. \end{array}\right.
$$

We now show how to solve the four subproblems in (4.3). The first subproblem in (4.3) corresponds to the proximal mapping of the nuclear norm function and is given by the matrix shrinkage operation

$$
L^{k+1} := \mathrm{MatShrink}(L^k - \tau(2L^k + S^k + Z^k - M - K^k - \mu\Lambda_1^k - \mu\Lambda_2^k), \mu\tau),
$$

where the matrix shrinkage operation $\text{MatShrink}(\cdot,\cdot)$ is defined in (2.12). The second subproblem in (4.3) is given by the ℓ_1 shrinkage operation

$$S^{k+1} := \text{Shrink}(S^k - \tau(L^k + S^k + Z^k - M - \mu\Lambda_1^k), \rho\mu\tau).$$

The third subproblem in (4.3) corresponds to projecting the matrix $W^k := M + \mu\Lambda_1^k - L^{k+1} - S^{k+1}$ onto the Euclidean ball $\|Z\|_F \le \sigma$ and this can be done by

$$Z^{k+1} := W^k / \max\{1, \|W^k\|_F/\sigma\}.$$

The fourth subproblem in (4.3) corresponds to projecting the matrix $L^{k+1} - \mu\Lambda_2^k$ onto the nonnegative orthant and this can be done by

$$K^{k+1} := \max\{L^{k+1} - \mu\Lambda_2^k, 0\},$$

where the max function is componentwise.

Application 3. APGM for TV-based MRI deblurring (2.22). We group $[y; z_1, \ldots, z_N]$ as one block of variable, so the constraint becomes (2.25). Then the augmented Lagrangian function of Problem (2.24) can be written as

$$\begin{array}{rl} \mathcal{L}_\mu(z,y,x;\lambda) := & \sum_i \|z_i\|_2 + \alpha\|Wx\|_1 + \mathcal{I}(\|y\|_2 \le \sigma) - \langle\gamma, Rx - y - b\rangle + \frac{1}{2\mu}\|Rx - y - b\|_2^2 \\ & - \sum_i \langle\lambda_i, D_i x - z_i\rangle + \frac{1}{2\mu}\sum_i \|D_i x - z_i\|_2^2, \end{array}$$

where $\lambda_i, i = 1, \ldots, N$ and γ are the Lagrange multipliers. The APGM for solving (2.24) can be described as:

$$\text{(4.4)} \qquad \left\{ \begin{array}{rcl} z_i^{k+1} & := & \text{argmin}_{z_i}\ \|z_i\|_2 + \frac{1}{2\mu\tau_1}\|z_i - (z_i^k + \tau_1(D_i x^k - z_i^k - \mu\lambda_i^k))\|_2^2, \quad \forall i \\ y^{k+1} & := & \text{argmin}_y\ \mathcal{I}(\|y\|_2 \le \sigma) + \frac{1}{2\mu\tau_1}\|y - (y^k + \tau_1(Rx^k - y^k - b - \mu\gamma^k))\|_2^2 \\ x^{k+1} & := & \text{argmin}_x\ \alpha\|Wx\|_1 + \frac{1}{2\mu\tau_2}\|x - (x^k - \tau_2 G_x^k)\|_2^2 \\ \lambda_i^{k+1} & := & \lambda_i^k - (D_i x^{k+1} - z_i^{k+1})/\mu, \quad \forall i \\ \gamma^{k+1} & := & \gamma^k - (Rx^{k+1} - y^{k+1} - b)/\mu, \end{array} \right.$$

where

$$G_x^k := (\sum_i D_i^\top D_i + R^\top R)x^k - \sum_i D_i^\top(z_i^{k+1} + \mu\lambda_i^k) - R^\top(y^{k+1} + b + \mu\gamma^k).$$

We now show how to solve the subproblems in (4.4). The first subproblem in (4.4) corresponds to the ℓ_2-norm shrinkage operation

$$z_i^{k+1} := \text{Shrink}_2(z_i^k + \tau_1(D_i x^k - z_i^k - \mu\lambda_i^k), \mu\tau_1),$$

where the ℓ_2-norm shrinkage operator is defined as (see, e.g., [52])

$$\text{Shrink}_2(w, \xi) := \frac{w}{\|w\|_2}\max\{\|w\|_2 - \xi, 0\},$$

where we adopted the convention $0/0 = 0$. The second subproblem in (4.4) corresponds to the projection of the vector $w^k := y^k + \tau_1(Rx^k - y^k - b - \mu\gamma^k)$ onto the Euclidean ball $\{y : \|y\|_2 \le \sigma\}$. Thus the solution of the second subproblem is given by

$$y^{k+1} := \frac{w^k}{\|w^k\|_2} \max\{\sigma, \|w^k\|_2\},$$

where we again adopted the convention $0/0 = 0$. The third subproblem in (4.4) corresponds to the ℓ_1 shrinkage operation and two wavelet transforms because the wavelet transform W is orthonormal:

$$x^{k+1} := W^\top \text{Shrink}(W(x^k - \tau_2 G_x^k), \alpha\mu\tau_2).$$

Thus we have shown that all the subproblems in (4.4) have closed-form solutions and can be solved relatively easily.

Application 4. APGM for CPCP (2.26). APGM for solving (2.26) can be described as follows.

$$\left\{\begin{array}{lcl} L^{k+1} & := & \text{argmin}_L \ \|L\|_* + \frac{1}{2\mu\tau_1}\|L - (L^k - \tau_1\mathcal{A}^*(\mathcal{A}(L^k + S^k) - M - \mu\Lambda^k))\|_F^2 \\ S^{k+1} & := & \text{argmin}_S \ \rho\|S\|_1 + \frac{1}{2\mu\tau_2}\|S - (S^k - \tau_2\mathcal{A}^*(\mathcal{A}(L^{k+1} + S^k) - M - \mu\Lambda^k))\|_F^2 \\ \Lambda^{k+1} & := & \Lambda^k - (\mathcal{A}(L^{k+1} + S^{k+1}) - M)/\mu. \end{array}\right. \tag{4.5}$$

Note that the two subproblems in (4.5) are both easy to solve. The solution of the first subproblem corresponds to the proximal mapping of the nuclear norm, which is given by

$$L^{k+1} := \text{MatShrink}(L^k - \tau_1\mathcal{A}^*(\mathcal{A}(L^k + S^k) - M - \mu\Lambda^k), \mu\tau_1).$$

The solution of the second subproblem corresponds to the proximal mapping of the ℓ_1 norm, which is given by

$$S^{k+1} := \text{Shrink}(S^k - \tau_2\mathcal{A}^*(\mathcal{A}(L^{k+1} + S^k) - M - \mu\Lambda^k), \mu\tau_2).$$

Application 5. APGM for Robust Alignment for Linearly Correlated Images (2.28). In the robust alignment problem (2.28), there are three blocks of variables. Here we group L and S as one big block $[L; S]$. The APGM for solving (2.28) can be described as

$$\left\{\begin{array}{lcl} L^{k+1} & := & \text{argmin}_L \ \|L\|_* + \frac{1}{2\mu\tau_1}\|L - (L^k - \tau_1(L^k + S^k - \hat{D}\Delta x^k - \mu\Lambda^k))\|_F^2 \\ S^{k+1} & := & \text{argmin}_S \ \rho\|S\|_1 + \frac{1}{2\mu\tau_1}\|S - (S^k - \tau_1(L^k + S^k - \hat{D}\Delta x^k - \mu\Lambda^k))\|_F^2 \\ \Delta x^{k+1} & := & \text{argmin}_{\Delta x} \frac{1}{2\mu\tau_2}\|\Delta x - (\Delta x^k + \tau_2\hat{D}^\top(L^{k+1} + S^{k+1} - \hat{D}\Delta x^k - \mu\Lambda^k))\|_2^2 \\ \Lambda^{k+1} & := & \Lambda^k - (L^{k+1} + S^{k+1} - \hat{D}\Delta x^{k+1})/\mu. \end{array}\right. \tag{4.6}$$

We notice that all three subproblems in (4.6) are easy to solve. Specifically, the first subproblem is the proximal mapping of nuclear norm, the second subproblem is the proximal mapping of ℓ_1 norm and the third subproblem is trivial.

5. Numerical Results. In this section, we apply APGM to solve three problems disccsued above: LVGGMS (2.17), SPCP with nonnegative constraint (2.20) and CPCP (2.26). We will focus on comparing APGM with LAL (3.26). We will show through these examples that APGM allows larger step sizes as suggested by the theoretical convergence guarantees, and thus is usually much faster than LAL. Our codes

were written in MATLAB. All the numerical tests were conducted in MATLAB version 7.12.0 on a laptop with Intel Core I5 2.5 GHz CPU and 4GB of RAM.

5.1. Results on LVGGMS (2.17)**.** We randomly created test problems using a procedure proposed by [48] and [47] for the classical graphical lasso problems. Similar procedures were used by [51] and [34]. For a given number of observed variables n and a given number of latent variables n_h, we first created a sparse matrix $U \in \mathbb{R}^{(n+n_h)\times(n+n_h)}$ with sparsity around 10%, i.e., 10% of the entries are nonzeros. The nonzero entries were set to -1 or 1 with equal probability. Then we computed $K := (U * U^\top)^{-1}$ as the true covariance matrix. We then chose the submatrix of K, $\hat{S} := K(1:n, 1:n)$ as the ground truth matrix of the sparse matrix S and chose $\hat{L} := K(1:n, n+1:n+n_h)K(n+1:n+n_h, n+1:n+n_h)^{-1}K(n+1:n+n_h, 1:n)$ as the ground truth matrix of the low rank matrix L. We then drew $N = 5n$ iid vectors, $Y_1, \ldots, Y_N$, from the Gaussian distribution $\mathcal{N}(\mathbf{0}, (\hat{S} - \hat{L})^{-1})$ by using the *mvnrnd* function in MATLAB, and computed a sample covariance matrix of the observed variables $\Sigma_X := \frac{1}{N}\sum_{i=1}^N Y_i Y_i^\top$.

We compared APGM (4.1) with LAL (3.26) on these randomly created data with different α and β. For the step sizes of the proximal gradient steps, we chose the largest step sizes that were allowed by the theoretical convergence guarantees. Thus, we chose $\tau_1 = 1$ and $\tau_2 = 0.5$ in (4.1), and $\tau = 1/3$ in (3.26). We chose the penalty parameter $\mu = 1$ in both (4.1) and (3.26). Both APGM and LAL were terminated whenever the relative residual of the equality constraint is below some threshold, i.e.,

$$resid := \frac{\|R - S + L\|_F}{\max\{1, \|R\|_F, \|S\|_F, \|L\|_F\}} < \epsilon_r.$$

In our numerical tests, we chose $\epsilon_r = 10^{-5}$. The initial points were chosen as R and S being the identity matrices and L and Λ being the zero matrices. We reported the comparison results on objective function value, number of iterations, CPU time and *resid* in Table 5.1. All CPU times reported were in seconds. For each instance, we randomly created ten examples, so the results reported in Table 5.1 were averaged over ten runs.

From Table 5.1 we see that, for different α and β, APGM needed much fewer number of iterations and much less CPU times than LAL. Besides, we noticed that APGM always produced solutions with smaller objective function values compared with LAL.

5.2. Results on SPCP with nonnegative constraints (2.20)**.** In this section, we apply APGM to solve the SPCP with nonnegative constraints (2.20). We implemented APGM for two types of partition of the variables, (2.21) and (3.29). Here we denote APGM with partition (2.21) as APGM1, and APGM with partition (3.29) as APGM2. Notice that as we discussed before, APGM1 and APGM2 allow the same largest step size for the first subproblem ($\tau_1 = 1/2.618$), while the second subproblem in APGM1 can take larger step size ($\tau_2 = 1$) than the second subproblem in APGM2 ($\tau_2 = 1/2$). We randomly created some SPCP problems in the following manner. For given $n, r < n$, we generated the $n \times n$ rank-r matrix $L^* = R_1 R_2^\top$, where $R_1 \in \mathbb{R}^{n\times r}$ and $R_2 \in \mathbb{R}^{n\times r}$ are both random matrices with all entries uniformly distributed in $[0, 1]$. Note that L^* is also a componentwise nonnegative matrix and it is the low-rank matrix we want to recover. The support of the sparse matrix S^* was chosen uniformly at random, and the nonzero entries of S^* were drawn uniformly in the interval [-500,500]. The entries of matrix Z^* for noise were generated as iid Gaussian with standard deviation 10^{-4}. We then set $M := L^* + S^* + Z^*$. We chose $\rho := 1/\sqrt{n}$ as suggested in [5]. The initial points for both APGM1 and APGM2 were chosen as $L^0 = K^0 = -M, S^0 = Z^0 = 0, \Lambda_1^0 = \Lambda_2^0 = 0$.

TABLE 5.1
Comparison of APGM and LAL for LVGGMS on randomly created data

dim	APGM				LAL			
n	obj	iter	cpu	resid	obj	iter	cpu	resid
$\alpha = 0.005, \beta = 0.05$								
100	9.852220e+001	230	1.2	9.90e-006	9.855560e+001	403	2.1	9.82e-006
200	1.784996e+002	283	6.9	9.88e-006	1.787355e+002	469	11.3	9.88e-006
500	1.968728e+002	322	81.3	9.66e-006	1.990030e+002	496	123.7	9.83e-006
1000	-8.160560e+001	406	744.2	9.70e-006	-7.188429e+001	526	963.4	9.86e-006
$\alpha = 0.01, \beta = 0.05$								
100	1.052696e+002	247	1.3	9.82e-006	1.052872e+002	386	2.1	9.94e-006
200	1.938810e+002	255	6.2	9.81e-006	1.940004e+002	395	9.6	9.87e-006
500	2.328487e+002	385	109.3	9.59e-006	2.341172e+002	476	135.5	9.73e-006
1000	-3.683218e+001	417	799.1	9.65e-006	-2.868802e+001	566	1074.5	9.80e-006
$\alpha = 0.01, \beta = 0.1$								
100	1.063504e+002	203	1.1	9.88e-006	1.063619e+002	392	2.0	9.93e-006
200	1.943391e+002	238	5.9	9.92e-006	1.944111e+002	451	11.0	9.94e-006
500	2.538428e+002	246	66.8	9.79e-006	2.546381e+002	386	104.7	9.90e-006
1000	5.841179e+001	369	666.3	9.87e-006	6.186235e+001	504	911.4	9.89e-006

We terminated the codes when the residual

$$resid := \frac{\|L + S + Z - M\|_F}{\|M\|_F} < \epsilon_r. \tag{5.1}$$

In our experiments, we chose $\epsilon_r = 10^{-4}$. We denote $Rank_r := r/n$ so that the rank of L^* is $n * Rank_r$, and $Card_r := cardinality(S^*)/(n^2)$ so that the cardinality of S^* is $n^2 * Card_r$. For different instance of $Rank_r, Card_r$ and μ, we report the number of iterations, relative error of the low-rank matrix L (rel_L), relative error of the sparse matrix S (rel_S) and CPU times in Table 5.2, where the relative errors are defined as

$$rel_L := \frac{\|L - L^*\|_F}{\|L^*\|_F}, \quad rel_S := \frac{\|S - S^*\|_F}{\|S^*\|_F}.$$

All CPU times reported are in seconds. For each instance, we randomly created ten examples, so the results reported in Table 5.2 were averaged over ten runs.

From Table 5.2 we have the following observations. For both $Rank_r = 0.01, Card_r = 0.01$ and $Rank_r = 0.05, Card_r = 0.05$, the results of APGM1 were very consistent. For almost all the cases of APGM1 in Table 5.2 (except the last instance, for which neither APGM1 nor APGM2 achieved a good accuracy), rel_L achieved the order 10^{-3} or lower and rel_S achieved the order 10^{-4} or lower. However, when $\mu = 10$, APGM2 cannot achieve the same order of accuracy. For all the problems in Table 5.2 (except the last instance), APGM1 needed much fewer number of iterations and much less CPU times than APGM2 to meet the stopping criterion (5.1). These observations validate our discussions above that APGM1 is expected to be more efficient than APGM2 because APMG1 allows larger step size than APGM2 for the second subproblem.

5.3. Results on CPCP. In this section, we apply APGM and LAL to solve the CPCP (2.26) to further compare their performance. We created some synthetic data in the following manner. We generated the $n \times n$ rank-r matrix $L^* = R_1 R_2^\top$, where $R_1 \in \mathbb{R}^{n \times r}$ and $R_2 \in \mathbb{R}^{n \times r}$ are both random matrices with all entries

TABLE 5.2
Comparison of APGM1 and APGM2 for SPCP with nonnegative constraints on randomly created data

dim	APGM1				APGM2			
n	iter	rel_L	rel_S	cpu	iter	rel_L	rel_S	cpu
$Rank_r = 0.01, Card_r = 0.01, \mu = 1000$								
100	129	8.30e-003	2.91e-005	0.6	544	9.02e-003	2.18e-005	2.4
200	156	4.71e-003	2.43e-005	3.5	687	4.93e-003	1.92e-005	15.1
500	80	2.02e-003	3.15e-005	18.8	469	2.16e-003	1.79e-005	111.4
1000	47	9.43e-004	5.23e-005	76.6	250	1.12e-003	1.78e-005	407.0
$Rank_r = 0.01, Card_r = 0.01, \mu = 100$								
100	42	8.44e-003	3.47e-005	0.2	94	9.18e-003	1.95e-005	0.4
200	52	4.91e-003	1.97e-005	1.1	78	4.85e-003	1.68e-005	1.7
500	38	2.11e-003	2.83e-005	8.4	60	2.02e-003	1.45e-005	13.0
1000	33	4.97e-004	5.31e-005	52.0	68	9.38e-004	1.07e-005	107.2
$Rank_r = 0.01, Card_r = 0.01, \mu = 10$								
100	39	5.26e-003	7.54e-005	0.2	111	1.84e+000	2.05e-002	0.5
200	43	3.24e-003	6.67e-005	0.9	155	1.95e-001	3.95e-003	3.1
500	49	1.37e-003	7.12e-005	10.0	198	8.52e-002	3.98e-003	40.3
1000	70	1.65e-003	1.44e-004	108.9	238	4.11e-002	3.70e-003	372.8
$Rank_r = 0.05, Card_r = 0.05, \mu = 1000$								
100	438	5.19e-003	4.56e-005	1.9	750	5.21e-003	4.33e-005	3.3
200	254	2.64e-003	4.46e-005	5.3	368	2.65e-003	4.10e-005	7.5
500	121	1.05e-003	4.82e-005	25.5	142	1.05e-003	4.38e-005	29.8
1000	70	5.14e-004	5.23e-005	111.8	87	4.79e-004	4.29e-005	138.6
$Rank_r = 0.05, Card_r = 0.05, \mu = 100$								
100	65	5.07e-003	4.13e-005	0.3	75	5.15e-003	4.06e-005	0.3
200	39	2.26e-003	5.59e-005	0.8	67	2.41e-003	3.04e-005	1.4
500	32	6.48e-004	6.68e-005	6.3	72	1.02e-003	3.31e-005	14.5
1000	45	3.53e-004	7.20e-005	71.9	74	5.27e-004	3.44e-005	118.5
$Rank_r = 0.05, Card_r = 0.05, \mu = 10$								
100	50	3.80e-003	7.91e-005	0.2	196	6.44e-002	1.34e-003	0.9
200	56	2.02e-003	7.93e-005	1.1	241	4.09e-002	1.63e-003	4.8
500	100	1.86e-003	1.74e-004	21.1	320	1.88e-002	1.85e-003	68.3
1000	95	4.89e-001	9.55e-002	151.2	27	1.31e+000	2.55e-001	44.9

uniformly distributed in $[0, 1]$. The support of the sparse matrix S^* was chosen uniformly at random, and the nonzero entries of S^* were drawn uniformly in the interval [-50,50]. We denote the matrix representative of $\mathcal{A} : \mathbb{R}^{m\times n} \to \mathbb{R}^p$ as $A \in \mathbb{R}^{p\times mn}$. Note that A is defined as $A\text{vec}(X) := \mathcal{A}(X)$, where $X \in \mathbb{R}^{m\times n}$ and $\text{vec}(X)$ is an mn-dimensional vector that is obtained by stacking all the columns of X. The entries of A were drawn as iid Gaussian $\mathcal{N}(0, 1)$. We then normalized A such that the largest singular value of A is 1. This is just to simplify the comparison of APGM and LAL, both of which need the information of the largest singular value of A to determine the step size used for the proximal gradient step. M in CPCP (2.26) was created as $M = A(L^* + S^*)$. We compared the performance of APGM and LAL with different step sizes on the proximal gradient steps. In particular, we denote APGM1 as APGM with both $\tau_1 = \tau_2 = 1/2$ in (4.5) and APGM2 as APGM with both $\tau_1 = \tau_2 = 1$ in (4.5). We denote LAL1 as LAL (3.26) with $\tau = 1/2$ and LAL2 as LAL with $\tau = 1$. Note that $\tau = 1$ violated the requirement in the convergence guarantee of LAL. We included this comparison to show that although APGM allows a larger step size, LAL is not allowed to take the same step size. We chose $\rho = 1/\sqrt{n}$ as suggested in [54]. The initial points for both APGM and

LAL were chosen as $L^0 = K^0 = -M, S^0 = Z^0 = 0, \Lambda_1^0 = \Lambda_2^0 = 0$. We terminated both APGM and LAL whenever the residual

$$\|\mathcal{A}(L+S) - M\|_F / \|M\|_F < \epsilon_r,$$

or it diverged immediately. In our experiments, we chose $\epsilon_r = 10^{-7}$. For different $n, Rank_r, Card_r, Samp_r, \mu$, we report the number of iterations, rel_L, rel_S and CPU times in Table 5.3, where $Rank_r$ and $Card_r$ are defined the same way as in Section 5.2, and $Samp_r := p/n^2$. In the experiments, we fixed $Rank_r = 0.02, Card_r = 0.02, Samp_r = 0.25$. We compared the performance of APGM1, APGM2, LAL1 and LAL2 for different n and μ. We tested four cases with $n = 50$ and 100 and $\mu = 1$ and 10 and reported the results in Table 5.3. For each instance, we randomly created ten problems, so the results reported in Table 5.3 were averaged over ten runs. "—" means that the algorithm diverged immediately.

From Table 5.3 we have the following observations. LAL worked well when $\tau = 1/2$. However, when $\tau = 1$, LAL only successfully solved the problems when $n = 50, \mu = 10$ and $n = 100, \mu = 10$. For the other two cases $n = 50, \mu = 1$ and $n = 100, \mu = 1$, LAL diverged immediately because the choice of $\tau = 1$ violated the requirement of the convergence guarantee. For APGM, we see that it successfully solved all the problems with $\tau_1 = \tau_2 = 1/2$ and $\tau_1 = \tau_2 = 1$. We also noticed that when APGM and LAL took the same step sizes and when LAL converged, LAL needed slightly more number of iterations but slightly less CPU times. This is because APGM is a Gauss-Seidel like method and thus is expected to be faster than the Jacobi like method LAL. However, the CPU times of LAL were less because the proximal gradient needed only to be computed once, this was one advantage of the Jacobi like method. Based on these observations, we can conclude that APGM is more stable with large step sizes, while LAL does not allow large step sizes and thus sometimes fails when large step sizes are taken. When APGM and LAL take their largest step sizes allowed, APGM is much faster than LAL.

6. Conclusions. Many modern applications arising from machine learning, statistics, computer vision etc. can be formulated as minimizing the sum of separable convex functions with linear linking constraints. Although the classical alternating direction method of multipliers can be used to solve these problems, its convergence result was ambiguous for problems with more than two blocks of variables and objective function. Besides, for many applications, the subproblems are not easy to solve. In this paper, we proposed an alternating proximal gradient method for solving structured convex optimization problems. Our method can be applied to solve problems with more than two separable blocks of variables and objective function. All the subproblems are easy to solve because their solutions correspond to the proximal mappings of the involved convex functions. The global convergence result of the proposed method was established. We compared the proposed method with the linearized augmented Lagrangian method on different applications. We showed through numerical experiments that the proposed alternating proximal gradient method allows larger step size on the proximal gradient step and thus is usually much faster than the linearized augmented Lagrangian method. We also illustrated through numerical examples how to group the variables into two blocks in order to use the proposed alternating proximal gradient method effectively.

Acknowledgement. The author is grateful to Professor Wotao Yin for reading an earlier version of this paper and for valuable suggestions and comments.

REFERENCES

TABLE 5.3
Results of APGM and LAL for CPCP

alg.	rel_L	rel_S	iter	cpu
$n = 50, \mu = 1$				
APGM1	5.92e-007	1.11e-007	1470	12.3
APGM2	5.72e-007	1.13e-007	735	6.1
LAL1	6.03e-007	1.13e-007	1488	8.3
LAL2	—	—	—	—
$n = 50, \mu = 10$				
APGM1	4.38e-007	1.13e-007	487	4.0
APGM2	3.63e-007	1.08e-007	277	2.3
LAL1	4.24e-007	1.12e-007	488	2.6
LAL2	3.73e-007	1.05e-007	279	1.5
$n = 100, \mu = 1$				
APGM1	4.63e-007	1.20e-007	1934	211.1
APGM2	4.56e-007	1.13e-007	973	106.0
LAL1	4.67e-007	1.22e-007	1959	131.6
LAL2	—	—	—	—
$n = 100, \mu = 10$				
APGM1	3.97e-007	1.18e-007	668	71.1
APGM2	3.75e-007	1.18e-007	406	43.4
LAL1	3.96e-007	1.19e-007	673	44.4
LAL2	3.77e-007	1.20e-007	411	27.0

[1] O. BANERJEE, L. EL GHAOUI, AND A. D'ASPREMONT, *Model selection through sparse maximum likelihood estimation for multivariate gaussian for binary data*, Journal of Machine Learning Research, 9 (2008), pp. 485–516.

[2] D. BOLEY, *Linear convergence of admm on a model problem*, tech. report, TR 12-009, Department of Computer Science and Engineering, University of Minnesota, 2012.

[3] S. BOYD, N. PARIKH, E. CHU, B. PELEATO, AND J. ECKSTEIN, *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Foundations and Trends in Machine Learning, (2011).

[4] J. CAI, E. J. CANDÈS, AND Z. SHEN, *A singular value thresholding algorithm for matrix completion*, SIAM J. on Optimization, 20 (2010), pp. 1956–1982.

[5] E. J. CANDÈS, X. LI, Y. MA, AND J. WRIGHT, *Robust principal component analysis?*, Journal of ACM, 58 (2011), pp. 1–37.

[6] E. J. CANDÈS AND B. RECHT, *Exact matrix completion via convex optimization*, Foundations of Computational Mathematics, 9 (2009), pp. 717–772.

[7] E. J. CANDÈS, J. ROMBERG, AND T. TAO, *Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information*, IEEE Transactions on Information Theory, 52 (2006), pp. 489–509.

[8] E. J. CANDÈS AND T. TAO, *The power of convex relaxation: near-optimal matrix completion*, IEEE Trans. Inform. Theory, 56 (2009), pp. 2053–2080.

[9] V. CHANDRASEKARAN, P.A. PARRILO, AND A.S. WILLSKY, *Latent variable graphical model selection via convex optimization*, preprint, (2010).

[10] V. CHANDRASEKARAN, S. SANGHAVI, P. PARRILO, AND A. WILLSKY, *Rank-sparsity incoherence for matrix decomposition*, SIAM Journal on Optimization, 21 (2011), pp. 572–596.

[11] G. CHEN AND M. TEBOULLE, *A proximal-based decomposition method for convex minimization problems*, Mathematical Programming, 64 (1994), pp. 81–101.

[12] P. L. COMBETTES AND JEAN-CHRISTOPHE PESQUET, *A Douglas-Rachford splitting approach to nonsmooth convex variational signal recovery*, IEEE Journal of Selected Topics in Signal Processing, 1 (2007), pp. 564–574.

[13] P. L. COMBETTES AND V. R. WAJS, *Signal recovery by proximal forward-backward splitting*, SIAM Journal on Multiscale Modeling and Simulation, 4 (2005), pp. 1168–1200.

[14] A. D'ASPREMONT, L. EL GHAOUI, M. I. JORDAN, AND G. R. G. LANCKRIET, *A direct formulation for sparse pca using semidefinite programming*, SIAM Review, 49 (2007), pp. 434–448.

[15] W. DENG AND W. YIN, *On the global and linear convergence of the generalized alternating direction method of multipliers*,

tech. report, Rice University CAAM, Technical Report TR12-14, 2012.

[16] D. Donoho, *Compressed sensing*, IEEE Transactions on Information Theory, 52 (2006), pp. 1289–1306.

[17] J. Douglas and H. H. Rachford, *On the numerical solution of the heat conduction problem in 2 and 3 space variables*, Transactions of the American Mathematical Society, 82 (1956), pp. 421–439.

[18] J. Eckstein, *Splitting methods for monotone operators with applications to parallel optimization*, PhD thesis, Massachusetts Institute of Technology, 1989.

[19] ———, *Some saddle-function splitting methods for convex programming*, Optimization Methods and Software, 4 (1994), pp. 75–83.

[20] J. Eckstein and D. P. Bertsekas, *On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators*, Math. Program., 55 (1992), pp. 293–318.

[21] J. Friedman, T. Hastie, and R. Tibshirani, *Sparse inverse covariance estimation with the graphical lasso*, Biostatistics, (2007).

[22] D. Gabay, *Applications of the method of multipliers to variational inequalities*, in Augmented Lagrangian Methods: Applications to the Solution of Boundary Value Problems, M. Fortin and R. Glowinski, eds., North-Hollan, Amsterdam, 1983.

[23] R. Glowinski and P. Le Tallec, *Augmented Lagrangian and Operator-Splitting Methods in Nonlinear Mechanics*, SIAM, Philadelphia, Pennsylvania, 1989.

[24] D. Goldfarb and S. Ma, *Fast multiple splitting algorithms for convex optimization*, SIAM Journal on Optimization, 22 (2012), pp. 533–556.

[25] D. Goldfarb, S. Ma, and K. Scheinberg, *Fast alternating linearization methods for minimizing the sum of two convex functions*, tech. report, Department of IEOR, Columbia University. Preprint available at http://arxiv.org/abs/0912.4571, 2010.

[26] ———, *Fast alternating linearization methods for minimizing the sum of two convex functions*, Mathematical Programming Series A, to appear, (2012).

[27] T. Goldstein, B. O'Donoghue, and S. Setzer, *Fast alternating direction optimization methods*, UCLA CAM Report 12-35, (2012).

[28] T. Goldstein and S. Osher, *The split Bregman method for L1-regularized problems*, SIAM J. Imaging Sci., 2 (2009), pp. 323–343.

[29] B. He, M. Tao, and X. Yuan, *Alternating direction method with gaussian back substitution for separable convex programming*, SIAM Journal on Optimization, 22 (2012), pp. 313–340.

[30] B. He and X. Yuan, *On the o(1/n) convergence rate of douglas-rachford alternating direction method*, SIAM Journal on Numerical Analysis, 50 (2012), pp. 700–709.

[31] B. S. He, L.-Z. Liao, D. Han, and H. Yang, *A new inexact alternating direction method for monotone variational inequalities*, Math. Program., 92 (2002), pp. 103–118.

[32] R. H. Keshavan, A. Montanari, and S. Oh, *Matrix completion from a few entries*, IEEE Trans. on Info. Theory, 56 (2010), pp. 2980–2998.

[33] S. Lauritzen, *Graphical Models*, Oxford University Press, 1996.

[34] L. Li and K.-C. Toh, *An inexact interior point method for l_1-regularized sparse covariance selection*, preprint, (2010).

[35] P. L. Lions and B. Mercier, *Splitting algorithms for the sum of two nonlinear operators*, SIAM Journal on Numerical Analysis, 16 (1979), pp. 964–979.

[36] Z. Q. Luo, *On the linear convergence of alternating direction method of multipliers*, preprint, (2012).

[37] M. Lustig, D. Donoho, and J. Pauly, *Sparse MRI: The application of compressed sensing for rapid MR imaging*, Magnetic Resonance in Medicine, in press (2007).

[38] S. Ma, *Alternating direction method of multipliers for sparse principal component analysis*, preprint, (2011).

[39] S. Ma, D. Goldfarb, and L. Chen, *Fixed point and Bregman iterative methods for matrix rank minimization*, Mathematical Programming Series A, 128 (2011), pp. 321–353.

[40] S. Ma, W. Yin, Y. Zhang, and A. Chakraborty, *An efficient algorithm for compressed MR imaging using total variation and wavelets*, IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), (2008), pp. 1–8.

[41] J. Malick, J. Povh, F. Rendl, and A. Wiegele, *Regularization methods for semidefinite programming*, SIAM Journal on Optimization, 20 (2009), pp. 336–356.

[42] N. Parikh and S. Boyd, *Block splitting for large-scale distributed learning*, in NIPS, 2011.

[43] D. H. Peaceman and H. H. Rachford, *The numerical solution of parabolic elliptic differential equations*, SIAM Journal on Applied Mathematics, 3 (1955), pp. 28–41.

[44] Y. Peng, A. Ganesh, J. Wright, W. Xu, and Y. Ma, *Rasl: Robust alignment by sparse and low-rank decomposition*

for linearly correlated images, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), (2012).

[45] Z. Qin, D. Goldfarb, and S. Ma, *An alternating direction method for total variation denoising*, preprint, (2011).

[46] B. Recht, M. Fazel, and P. Parrilo, *Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization*, SIAM Review, 52 (2010), pp. 471–501.

[47] K. Scheinberg, S. Ma, and D. Goldfarb, *Sparse inverse covariance selection via alternating linearization methods*, in Proceedings of the Neural Information Processing Systems (NIPS), 2010.

[48] K. Scheinberg and I. Rish, *Sinco - a greedy coordinate ascent method for sparse inverse covariance selection problem*, (2009). Preprint available at http://www.optimization-online.org/DB_HTML/2009/07/2359.html.

[49] M. Tao and X. Yuan, *Recovering low-rank and sparse components of matrices from incomplete and noisy observations*, SIAM J. Optim., 21 (2011), pp. 57–81.

[50] R. Tibshirani, *Regression shrinkage and selection via the lasso*, J. Royal. Statist. Soc B., 58 (1996), pp. 267–288.

[51] C. Wang, D. Sun, and K.-C. Toh, *Solving log-determinant optimization problems by a Newton-CG primal proximal point algorithm*, preprint, (2009).

[52] Y. Wang, J. Yang, W. Yin, and Y. Zhang, *A new alternating minimization algorithm for total variation image reconstruction*, SIAM Journal on Imaging Sciences, 1 (2008), pp. 248–272.

[53] Z. Wen, D. Goldfarb, and W. Yin, *Alternating direction augmented Lagrangian methods for semidefinite programming*, Mathematical Programming Computation, 2 (2010), pp. 203–230.

[54] J. Wright, A. Ganesh, K. Min, and Y. Ma, *Compressive principal component pursuit*, preprint available at http://arxiv.org/pdf/1202.4596v1.pdf, (2012).

[55] J. Yang and X. Yuan, *Linearized augmented lagrangian and alternating direction methods for nuclear norm minimization*, Mathematics of Computation, (2011).

[56] J. Yang and Y. Zhang, *Alternating direction algorithms for ℓ_1 problems in compressive sensing*, SIAM Journal on Scientific Computing, 33 (2011), pp. 250–278.

[57] J. Yang, Y. Zhang, and W. Yin, *A fast TVL1-L2 algorithm for image reconstruction from partial fourier data*, IEEE Journal of Selected Topics in Signal Processing Special Issue on Compressed Sensing., 4 (2010), pp. 288–297.

[58] M. Yuan and Y. Lin, *Model selection and estimation in the Gaussian graphical model*, Biometrika, 94 (2007), pp. 19–35.

[59] X. Yuan, *Alternating direction methods for sparse covariance selection*, Journal of Scientific Computing, (2009).

[60] X. Zhang, M. Burger, X. Bresson, and S. Osher, *Bregmanized nonlocal regularization for deconvolution and sparse reconstruction*, SIAM Journal on Imaging Science, 3 (2010), p. 253276.

[61] Z. Zhou, X. Li, J. Wright, E. J. Candès, and Y. Ma, *Stable principal component pursuit*, Proceedings of International Symposium on Information Theory, (2010).

[62] H. Zou, T. Hastie, and R. Tibshirani, *Sparse principle component analysis*, J. Comput. Graph. Stat., 15 (2006), pp. 265–286.