JORC (2013) 1:253–274
DOI 10.1007/s40305-013-0016-9

Alternating Direction Method of Multipliers for Sparse Principal Component Analysis

Shiqian Ma

Received: 11 January 2013 / Revised: 23 May 2013 / Accepted: 27 May 2013 /
Published online: 8 June 2013

Abstract We consider a convex relaxation of sparse principal component analysis proposed by d'Aspremont et al. (SIAM Rev. 49:434–448, 2007). This convex relaxation is a nonsmooth semidefinite programming problem in which the ℓ_1 norm of the desired matrix is imposed in either the objective function or the constraint to improve the sparsity of the resulting matrix. The sparse principal component is obtained by a rank-one decomposition of the resulting sparse matrix. We propose an alternating direction method based on a variable-splitting technique and an augmented Lagrangian framework for solving this nonsmooth semidefinite programming problem. In contrast to the first-order method proposed in d'Aspremont et al. (SIAM Rev. 49:434–448, 2007), which solves approximately the dual problem of the original semidefinite programming problem, our method deals with the primal problem directly and solves it exactly, which guarantees that the resulting matrix is a sparse matrix. A global convergence result is established for the proposed method. Numerical results on both synthetic problems and the real applications from classification of text data and senate voting data are reported to demonstrate the efficacy of our method.

Keywords Sparse PCA · Semidefinite programming · Alternating direction method · Augmented Lagrangian method · Deflation · Projection onto the simplex

Mathematics Subject Classification 62H25 · 90C25 · 90C22 · 62H30 · 65K05

1 Introduction

Principal component analysis (PCA) plays an important role in applications arising from data analysis, dimension reduction and bioinformatics etc. PCA finds a few

S. Ma (✉)
Department of Systems Engineering and Engineering Management, The Chinese University of Hong Kong, Shatin, NT, Hong Kong
e-mail: sqma@se.cuhk.edu.hk

linear combinations of the original variables. These linear combinations, which are called principal components (PCs), are orthogonal to each other and explain most of the variance of the data. Specifically, let $\xi = (\xi_1, \cdots, \xi_p)$ be a p-dimensional random vector. Then for a given data matrix $M \in \mathbb{R}^{p\times n}$, which consists of n samples of the p variables, PCA corresponds to a singular value decomposition (SVD) of M or an eigenvalue decomposition of a sample covariance matrix Σ. Without loss of generality, assuming M is centered, i.e., the means of the rows of X are all zeros, then a commonly used sample covariance matrix is $\Sigma = MM^\top/(n-1)$. Assume the eigenvalue decomposition of Σ is given by $\Sigma = V\Lambda V^\top$, then the columns of ξV are the PCs and the columns of V are called loading vectors of the corresponding PCs. Thus, finding the PC that explains the largest variance of the variables corresponds to the following eigenvalue problem:

$$x^* := \arg\max x^\top \Sigma x, \quad \text{s.t. } \|x\|_2 \leqslant 1, \tag{1.1}$$

where $\|x\|_2$ is the Euclidean norm of vector x. Problem (1.1) actually gives the eigenvector that corresponds to the largest eigenvalue of Σ. However, the loading vector x^* is not expected to have many zero coefficients. This makes it hard to explain the PC. For example, in the text classification problem, we are given a binary data matrix $M \in \mathbb{R}^{p\times n}$ that records the occurrences of p words in n postings. That is, $M_{ij} = 1$ if the ith word appears in the jth posting and $M_{ij} = 0$ if the ith word does not appear in the jth posting. The standard PCA cannot tell which words contribute most to the explained variance since the loadings are linear combinations of all the variables. Thus, sparse PCs are needed because it is easier to analyze which variables contribute most to the explained variance.

Many techniques were proposed to extract sparse PCs from given sample covariance matrix Σ or sample data matrix M. One natural thought is to impose a cardinality constraint to (1.1), which leads to the following formulation for sparse PCA:

$$x^* := \arg\max x^\top \Sigma x, \quad \text{s.t. } \|x\|_2 \leqslant 1, \quad \|x\|_0 \leqslant K, \tag{1.2}$$

where $\|x\|_0$ (the ℓ_0 norm of x) counts the number of nonzeros of x and the integer K controls the sparsity of the solution. Note that the cardinality constraint $\|x\|_0 \leqslant K$ makes the problem nonconvex, which is usually numerically challenging to solve. Some other nonconvex models and algorithms for solving them are also considered in [8, 21, 23, 24, 42]. It should be pointed out that although these algorithms are quite efficient, the convergence results are usually not very strong. Especially, there is no result for global convergence as the models are nonconvex.

In this paper, we will focus on a convex relaxation of (1.2) that was proposed by d'Aspremont et al. in [9]. The convex relaxation in [9] is a semidefinite programming (SDP) problem based on the lifting and projection technique, which is a standard technique in using SDP to approximate combinatorial optimization problems (see e.g., [1, 3, 34]). Note that if we denote $X = xx^\top$, then (1.2) can be rewritten as

$$\max_{X\in\mathbb{R}^{p\times p}} \left\{ \langle \Sigma, X\rangle, \ \text{s.t. } \mathbf{Tr}(X) = 1, \|X\|_0 \leqslant K^2, X \succeq 0, \operatorname{rank}(X) = 1 \right\}, \tag{1.3}$$

where $\mathbf{Tr}(X)$ denotes the trace of matrix X. The rank constraint is then dropped and the cardinality constraint is replaced by ℓ_1 norm constraint, and this leads to the

following convex problem, which is an SDP.

$$\max_{X\in\mathbb{R}^{p\times p}} \big\{\langle \Sigma, X\rangle,\ \text{s.t. } \mathbf{Tr}(X)=1, \|X\|_1 \leqslant K, X\succeq 0\big\}, \tag{1.4}$$

where the ℓ_1 norm of X is defined as $\|X\|_1 := \sum_{ij}|X_{ij}|$ and $\|X\|_1 \leqslant K$ is imposed to promote the sparsity of the solution. This is inspired by the recent emergence of compressed sensing (see e.g., [5, 10]). Note that $\|X\|_1 \leqslant K$ is used in (1.4) instead of $\|X\|_1 \leqslant K^2$. This is due to the fact that, when $X = xx^\top$ and $\mathbf{Tr}(X)=1$, we have $\|X\|_F = 1$, and also that if $\|X\|_0 \leqslant K^2$, then $\|X\|_1 \leqslant K\|X\|_F$. After the optimal solution X^* to (1.4) is obtained, the vector $\hat{x}$ from the rank-one decomposition of X^*, i.e., $X^* = \hat{x}\hat{x}^\top$ is used as an approximation of the solution of (1.2). This is the whole procedure of the lifting and projection technique. Although some standard methods such as interior point methods can be used to solve the SDP (1.4) (see e.g., [1, 3, 34]), it is not wise to do so because (1.4) is a nonsmooth problem, and transforming it to a standard SDP increases the size of the problem dramatically.

It is known that (1.4) is equivalent to the following problem with an appropriately chosen parameter $\rho > 0$:

$$\max_{X\in\mathbb{R}^{p\times p}} \big\{\langle \Sigma, X\rangle - \rho\|X\|_1 \text{ s.t. } \mathbf{Tr}(X)=1, X\succeq 0\big\}. \tag{1.5}$$

Note that (1.5) can be rewritten as

$$\max_{X\succeq 0, \mathbf{Tr}(X)=1} \min_{\|U\|_\infty\leqslant\rho} \langle \Sigma + U, X\rangle, \tag{1.6}$$

where $\|U\|_\infty$ denotes the largest component of U in magnitude, i.e., $\|U\|_\infty = \max_{ij}|U_{ij}|$. The dual problem of (1.5) is given by interchanging the max and min in (1.6), i.e.,

$$\min_{\|U\|_\infty\leqslant\rho} \max_{X\succeq 0, \mathbf{Tr}(X)=1} \langle \Sigma + U, X\rangle,$$

which can be further reduced to

$$\min_{U\in\mathbb{R}^{p\times p}} \lambda_{\max}(\Sigma + U), \quad \text{s.t. } \|U\|_\infty \leqslant \rho, \tag{1.7}$$

where $\lambda_{\max}(Z)$ denotes the largest eigenvalue of matrix Z. d'Aspremont et al. [9] proposed to solve the dual problem (1.7) using Nesterov's first-order algorithm (see e.g., [27, 28]), which is an accelerated projected gradient method. However, since the objective function of (1.7) is nonsmooth, one needs to smooth it in order to apply Nesterov's algorithm. Thus, the authors of [9] actually solve an approximation of the dual problem (1.7), which can be formulated as follows.

$$\min f_\mu(U), \quad \text{s.t. } \|U\|_\infty \leqslant \rho, \tag{1.8}$$

where $\mu > 0$ is the smoothing parameter, $f_\mu(U) := \max\{\langle \Sigma + U, X\rangle - \mu d(X),\ \text{s.t. } \mathbf{Tr}(X)=1, X\succeq 0\}$ and $d(X) := \mathbf{Tr}(X\log X) + \log(n)$. It is shown in [9] that an approximate solution X^k to the primal problem (1.5) can be obtained by $X^k = \nabla f_\mu(U^k)$, where U^k is an approximate solution of (1.8). It is easy to see that X^k

is not guaranteed to be a sparse matrix. Besides, although there is no duality gap between (1.5) and (1.7), the authors solve an approximation of (1.7). It needs also to be noted that Nesterov's algorithm used in [9] cannot solve the constrained problem (1.4). Although (1.4) and (1.5) are equivalent with appropriately chosen parameters K and ρ, in many applications, it is usually easier to choose an appropriate K since we know how many nonzeros are preferred in the sparse PCs.

Note that (1.2) only gives the largest sparse PC. In many applications, several leading sparse PCs are needed in order to explain more variance. Multiple sparse PCs are usually found by solving a sequence of sparse PCA problems (1.2), with Σ constructed via the so-called *deflation* technique for each sparse PC.

In this paper, we propose an alternating direction method based on a variable-splitting technique and an augmented Lagrangian framework for solving directly the primal problems (1.4) and (1.5). Our method solves two subproblems in each iteration. One subproblem has a closed-form solution that corresponds to projecting a given matrix onto the simplex of the cone of semidefinite matrices. This projection requires an eigenvalue decomposition. The other subproblem has a closed-form solution that corresponds to a vector shrinkage operation (for Problem (1.5)) or a projection onto the ℓ_1 ball (for Problem (1.4)). Thus, our method produces two iterative points at each iteration. One iterative point is a semidefinite matrix with trace equal to one and the other one is a sparse matrix. Eventually these two points will converge to the same point, and thus we get an optimal solution which is a sparse and semidefinite matrix. Compared with Nesterov's first-order method suggested in [9] for solving the approximated dual problem (1.8), our method can solve the nonsmooth primal problems (1.4) and (1.5) uniformly. Also, since we deal with the primal problems directly, the ℓ_1 norm in the constraint or the objective function guarantees that our solution is a sparse matrix, while Nesterov's method in [9] does not guarantee this since it solves the approximated dual problem.

The rest of the paper is organized as follows. In Sect. 2, we introduce our alternating direction method of multipliers for solving the nonsmooth SDP problems (1.4) and (1.5). We discuss some practical issues including the deflation technique for computing multiple sparse PCs in Sect. 3. In Sect. 4, we use our alternating direction method of multipliers to solve sparse PCA problems arising from different applications such as classification of text data and senate voting records to demonstrate the efficacy of our method. We make some conclusions in Sect. 5.

2 Alternating Direction Method of Multipliers

We first introduce some notation. We use $\mathcal{C}$ to denote the simplex of the cone of the semidefinite matrices, i.e., $\mathcal{C} = \{X \in \mathbb{R}^{p\times p} \mid \mathbf{Tr}(X) = 1, X \succeq 0\}$. We use $\mathcal{B}$ to denote the ℓ_1-ball with radius K in $\mathbb{R}^{p\times p}$, i.e., $\mathcal{B} = \{X \in \mathbb{R}^{p\times p} \mid \|X\|_1 \leqslant K\}$. $I_{\mathcal{A}}(X)$ denotes the indicator function of set $\mathcal{A}$, i.e.,

$$I_{\mathcal{A}}(X) = \begin{cases} 0 & \text{if } X \in \mathcal{A}, \\ +\infty & \text{otherwise.} \end{cases} \tag{2.1}$$

We know that $I_{\mathcal{C}}(X)$ and $I_{\mathcal{B}}(X)$ are both convex functions since $\mathcal{C}$ and $\mathcal{B}$ are both convex sets. We then can reformulate (1.4) and (1.5) uniformly as the following unconstrained problem:

$$\min -\langle \Sigma, X\rangle + I_{\mathcal{C}}(X) + h(X), \tag{2.2}$$

where $h(X) = I_{\mathcal{B}}(X)$ for (1.4) and $h(X) = \rho\|X\|_1$ for (1.5). Note that $h(X)$ is convex in both cases. (2.2) can be also viewed as the following inclusion problem:

$$\text{Find } X, \quad \text{s.t. } 0 \in -\Sigma + \partial I_{\mathcal{C}}(X) + \partial h(X). \tag{2.3}$$

Problem (2.3) finds zero of the sum of two monotone operators. Methods based on operator-splitting techniques, such as Douglas–Rachford method and Peaceman–Rachford method, are usually used to solve Problem (2.3) (see e.g., [6, 7, 11, 13, 14, 22, 30]). From the convex optimization perspective, the alternating direction method of multipliers (ADMM) for solving (2.2) is a direct application of the Douglas–Rachford method. ADMM has been successfully used to solve structured convex optimization problems arising from image processing, compressed sensing, machine learning, semidefinite programming etc. (see e.g., [2, 15–19, 26, 33, 36–39]). We now show how ADMM can be used to solve the sparse PCA problem (2.2).

ADMM for solving (2.2) is based on a variable-splitting technique and an augmented Lagrangian framework. By introducing a new variable Y, (2.2) can be rewritten as

$$\begin{aligned} &\min -\langle \Sigma, X\rangle + I_{\mathcal{C}}(X) + h(Y) \\ &\text{s.t. } X = Y. \end{aligned} \tag{2.4}$$

Note that although the number of variables is increased, the two nonsmooth functions $I_{\mathcal{C}}(\cdot)$ and $h(\cdot)$ are now separated since they are associated with different variables. For this equality-constrained problem, augmented Lagrangian method is a standard approach to solve it. A typical iteration of augmented Lagrangian method for solving (2.4) is given by

$$\begin{cases} \big(X^{k+1}, Y^{k+1}\big) := \arg\min\limits_{(X,Y)} \mathcal{L}_{\mu}\big(X, Y; \Lambda^k\big), \\ \Lambda^{k+1} := \Lambda^k - \dfrac{1}{\mu}\big(X^{k+1} - Y^{k+1}\big), \end{cases} \tag{2.5}$$

where the augmented Lagrangian function $\mathcal{L}_{\mu}(X, Y; \Lambda)$ is defined as:

$$\mathcal{L}_{\mu}(X, Y; \Lambda) := -\langle \Sigma, X\rangle + I_{\mathcal{C}}(X) + h(Y) - \langle \Lambda, X - Y\rangle + \frac{1}{2\mu}\|X - Y\|_F^2, \tag{2.6}$$

where $\mu > 0$ is a penalty parameter and Λ is the Lagrange multiplier associated with the linear constraint $X = Y$. Note that it is usually hard to minimize the augmented Lagrangian function $\mathcal{L}_{\mu}(X, Y; \Lambda^k)$ with respect to X and Y simultaneously. In fact, it is as difficult as solving the original problem (2.4). However, if we minimize the augmented Lagrangian function with respect to X and Y alternatingly, we obtain two subproblems in each iteration and both of them are relatively easy to solve. This

results in the following alternating direction method of multipliers:

$$\begin{cases} X^{k+1} := \arg\min_X \mathcal{L}_\mu\big(X, Y^k; \Lambda^k\big), \\ Y^{k+1} := \arg\min_Y \mathcal{L}_\mu\big(X^{k+1}, Y; \Lambda^k\big), \\ \Lambda^{k+1} := \Lambda^k - \big(X^{k+1} - Y^{k+1}\big)/\mu. \end{cases} \tag{2.7}$$

It can be shown that the two subproblems in (2.7) are both relatively easy to solve in the sparse PCA problem. Before we show that, we characterize two nice properties of the indicator function (2.1).

Property 1 The proximal mapping of the indicator function $I_{\mathcal{A}}(\cdot)$ is the Euclidean projection onto $\mathcal{A}$, i.e.,

$$\text{prox}_{I_{\mathcal{A}}}(X) \equiv \mathcal{P}_{\mathcal{A}}(X), \tag{2.8}$$

where

$$\text{prox}_{I_{\mathcal{A}}}(X) := \arg\min_U \left\{ I_{\mathcal{A}}(U) + \frac{1}{2}\|U - X\|_F^2 \right\}, \tag{2.9}$$

and

$$\mathcal{P}_{\mathcal{A}}(X) := \arg\min_U \left\{ \frac{1}{2}\|U - X\|_F^2, \text{ s.t. } U \in \mathcal{A} \right\}. \tag{2.10}$$

Property 2 The optimality conditions for Problem (2.10) are given by

$$X - U^* \in \partial I_{\mathcal{A}}\big(U^*\big), \tag{2.11}$$

which is equivalent to

$$\big\langle X - U^*, Z - U^* \big\rangle \leqslant 0, \quad \forall Z \in \mathcal{A}, \tag{2.12}$$

where U^* is the optimal solution of (2.10).

Now, the first subproblem in (2.7) can be reduced to

$$X^{k+1} := \arg\min \left\{ \mu I_{\mathcal{C}}(X) + \frac{1}{2}\big\| X - \big(Y^k + \mu\Lambda^k + \mu\Sigma\big)\big\|_F^2 \right\}, \tag{2.13}$$

which can be further reduced to projection onto $\mathcal{C}$ using Property 1,

$$X^{k+1} = \mathcal{P}_{\mathcal{C}}\big(Y^k + \mu\Lambda^k + \mu\Sigma\big) := \arg\min \left\{ \frac{1}{2}\big\| X - \big(Y^k + \mu\Lambda^k + \mu\Sigma\big)\big\|_F^2, \text{ s.t. } X \in \mathcal{C} \right\}. \tag{2.14}$$

When $h(Y) = I_{\mathcal{B}}(Y)$ as in Problem (1.4), the second subproblem in (2.7) can be reduced to

$$Y^{k+1} := \arg\min \left\{ \mu I_{\mathcal{B}}(Y) + \frac{1}{2}\big\| Y - \big(X^{k+1} - \mu\Lambda^k\big)\big\|_F^2 \right\}, \tag{2.15}$$

which can be further reduced to projection onto $\mathcal{B}$ using Property 1,

$$Y^{k+1} = \mathcal{P}_{\mathcal{B}}\big(X^{k+1} - \mu\Lambda^k\big) := \arg\min\left\{\frac{1}{2}\big\|Y - \big(X^{k+1} - \mu\Lambda^k\big)\big\|_F^2,\ \text{s.t. } Y \in \mathcal{B}\right\}. \tag{2.16}$$

When $h(Y) = \rho\|Y\|_1$ as in Problem (1.5), the second subproblem in (2.7) can be reduced to

$$Y^{k+1} := \arg\min\left\{\mu\rho\|Y\|_1 + \frac{1}{2}\big\|Y - \big(X^{k+1} - \mu\Lambda^k\big)\big\|_F^2\right\}. \tag{2.17}$$

Problem (2.17) has a closed-form solution that is given by

$$Y^{k+1} = \text{Shrink}\big(X^{k+1} - \mu\Lambda^k, \mu\rho\big), \tag{2.18}$$

where the shrinkage operator is defined as

$$\big(\text{Shrink}(Z,\tau)\big)_{ij} := \text{sgn}(Z_{ij})\max\big\{|Z_{ij}| - \tau, 0\big\}, \quad \forall i, j. \tag{2.19}$$

In the following, we will show that (2.13) and (2.15) are easy to solve, i.e., the two projections (2.14) and (2.16) can be done efficiently. First, since the problem of projection onto $\mathcal{C}$

$$\mathcal{P}_{\mathcal{C}}(X) = \arg\min\left\{\frac{1}{2}\|Z - X\|_F^2,\ \text{s.t. } \mathbf{Tr}(Z) = 1, Z \succeq 0\right\} \tag{2.20}$$

is unitary-invariant, its solution is given by $\mathcal{P}_{\mathcal{C}}(X) = U\,\text{diag}(\gamma)U^\top$, where $X = U\,\text{diag}(\sigma)U^\top$ is the eigenvalue decomposition of X, and γ is the projection of σ onto the simplex in the Euclidean space, i.e.,

$$\gamma := \arg\min\left\{\frac{1}{2}\|\xi - \sigma\|_2^2,\ \text{s.t. } \sum_{i=1}^p \xi_i = 1, \xi \geqslant 0\right\}. \tag{2.21}$$

We consider a slightly more general problem:

$$\xi^* := \arg\min\left\{\frac{1}{2}\|\xi - \sigma\|_2^2,\ \text{s.t. } \sum_{i=1}^p \xi_i = r, \xi \geqslant 0\right\}, \tag{2.22}$$

where scalar $r > 0$. Note that (2.21) is a special case of (2.22) with $r = 1$. From the first-order optimality conditions for (2.22), it is easy to show that the optimal solution of (2.22) is given by

$$\xi_i^* := \max\{\sigma_i - \theta, 0\}, \quad \forall i = 1, \cdots, p,$$

where the scalar θ is the solution of the following piecewise linear equation:

$$\sum_{i=1}^p \max\{\sigma_i - \theta, 0\} = r. \tag{2.23}$$

It is known that the piecewise linear equation (2.23) can be solved quite efficiently and thus solving (2.22) can be done easily. In fact, the following procedure (Algorithm 1) gives the optimal solution of (2.22). We refer the readers to [31] for the proof of the validity of the algorithm. It is easy to see that Algorithm 1 has an $O(p \log p)$ complexity. Linear time algorithms for solving (2.22) are studied in [4, 12, 29]. Thus, solving (2.13) corresponds to an eigenvalue decomposition and a projection onto the simplex in the Euclidean space, and they both can be done efficiently.

Algorithm 1: Projection onto the simplex in the Euclidean space

Input: A vector $\sigma \in \mathbb{R}^p$ and a scalar $r > 0$.
Sort σ into $\hat{\sigma}$ as a non-decreasing order: $\hat{\sigma}_1 \leqslant \hat{\sigma}_2 \leqslant \cdots \leqslant \hat{\sigma}_p$
Find index $\hat{j}$, the smallest j such that $\hat{\sigma}_j - \frac{1}{p-j+1}\big(\sum_{i=j}^{p} \hat{\sigma}_i - r\big) > 0$
Compute $\theta = \frac{1}{p-\hat{j}+1}\big(\sum_{i=\hat{j}}^{p} \hat{\sigma}_i - r\big)$
Output: A vector γ, s.t. $\gamma_i = \max\{\sigma_i - \theta, 0\}, i = 1, \cdots, p$.

Solving (2.15) (or equivalently (2.16)) corresponds to a projection onto the ℓ_1-ball: $\|Y\|_1 \leqslant K$. It has been shown in [12, 35] that projection onto the ℓ_1-ball can be done easily. In fact, the solution of

$$\hat{\gamma} = \arg\min\left\{\frac{1}{2}\|\xi - \hat{\sigma}\|_2^2,\ \text{s.t. } \|\xi\|_1 \leqslant r\right\} \tag{2.24}$$

is given by $\hat{\gamma}_i = \operatorname{sgn}(\hat{\sigma}_i)\gamma_i, \forall i = 1, \cdots, p$, where γ is the solution of

$$\min \frac{1}{2}\big\|\gamma - |\hat{\sigma}|\big\|_2^2, \quad \text{s.t. } \sum_{i=1}^{p} \gamma_i = r, \quad \gamma \geqslant 0,$$

i.e., the projection of $|\hat{\sigma}|$ (elementwise absolute value of $\hat{\sigma}$) onto the simplex. Thus, (2.15) can be rewritten as

$$\operatorname{vec}\big(Y^{k+1}\big) = \arg\min\left\{\frac{1}{2}\big\|y - \operatorname{vec}\big(X^{k+1} - \mu\Lambda^k\big)\big\|_2^2,\ \text{s.t. } \|y\|_1 \leqslant K\right\}, \tag{2.25}$$

and it corresponds to a projection onto the simplex in the Euclidean space, where $\operatorname{vec}(Y)$ denotes the vector form of Y which is obtained by stacking the columns of Y into a long vector.

To summarize, our ADMM for solving (1.4) and (1.5) can be uniformly described as Algorithm 2.

Remark 2.1 Although Algorithm 2 suggests that we need to compute the eigenvalue decomposition of $Y^k + \mu\Lambda^k + \mu\Sigma$ in order to get the solution to (2.13), we actually only need to compute the positive eigenvalues and corresponding eigenvectors of $Y^k + \mu\Lambda^k + \mu\Sigma$.

Algorithm 2: ADMM for solving (1.4) and (1.5)

Initialization: $Y^0 = 0$, $\Lambda^0 = 0$.
for $k = 0, 1, \cdots$ **do**
Compute the eigenvalue decomposition: $Y^k + \mu\Lambda^k + \mu\Sigma = U\,\mathrm{diag}(\sigma)U^\top$
Project σ onto the simplex in Euclidean space by Algorithm 1, and denote the solution by γ
Compute $X^{k+1} = U\,\mathrm{diag}(\gamma)U^\top$
Perform one of the following:

- if (1.4) is solved, update Y^{k+1} by solving (2.25)
- if (1.5) is solved, update Y^{k+1} by (2.18)

Update Λ^{k+1} by $\Lambda^{k+1} = \Lambda^k - (X^{k+1} - Y^{k+1})/\mu$

We have the following global convergence result for Algorithm 2.

Theorem 2.2 *The sequence* $\{(X^k, Y^k, \Lambda^k)\}$ *produced by* (2.7) (*Algorithm* 2) *from any starting point converges to an optimal solution to Problem* (2.4).

Proof The proof of this convergence result is a direct application of the well studied convergence result for Douglas–Rachford operator splitting method (see [13, 14]). We include the specialized proof for Problem (2.4) in the Appendix just for the sake of completeness. □

3 The Deflation Techniques and Other Practical Issues

It should be noticed that the solution of Problem (1.1) only gives the largest eigenvector (the eigenvector corresponding to the largest eigenvalue) of Σ. In many applications, the largest eigenvector is not enough to explain the total variance of the data. Thus one usually needs to compute several leading eigenvectors to explain more variance of the data. Hotelling's deflation method [32] is usually used to extract the leading eigenvectors sequentially. The Hotelling's deflation method extracts the rth leading eigenvector of Σ by solving

$$x_r = \arg\max\left\{x^\top \Sigma_{r-1} x,\ \text{s.t. } \|x\|_2 \leqslant 1\right\},$$

where $\Sigma_0 := \Sigma$ and

$$\Sigma_r = \Sigma_{r-1} - x_r x_r^\top \Sigma_{r-1} x_r x_r^\top.$$

It is easy to verify that Hotelling's deflation method preserves the positive-semidefiniteness of matrix Σ_r. However, as pointed out in [25], it does not preserve the positive-semidefiniteness of Σ_r when it comes to the sparse PCA problem (1.2), because the solution x_r is no longer an eigenvector of Σ_{r-1}. Thus, the second leading eigenvector produced by solving the sparse PCA problem may not explain

well the variance of the data. We should point out that the deflation method used in [9] is Hotelling's deflation method.

Several deflation techniques to overcome this difficulty for sparse PCA were proposed by Mackey in [25]. In our numerical experiments, we chose to use the Schur complement deflation method in [25]. The Schur complement deflation method updates matrix Σ_r by

$$\Sigma_r = \Sigma_{r-1} - \frac{\Sigma_{r-1} x_r x_r^\top \Sigma_{r-1}}{x_r^\top \Sigma_{r-1} x_r}. \tag{3.1}$$

The Schur complement deflation method has the following properties as shown in [25]. (i) Schur complement deflation preserves the positive-semidefiniteness of Σ_r, i.e., $\Sigma_r \succeq 0$. (ii) Schur complement deflation renders x_s orthogonal to Σ_r for $s \leqslant r$, i.e., $\Sigma_r x_s = 0,\ \forall s \leqslant r$.

When we want to find the leading r sparse PCs of Σ, we use ADMM to solve sequentially r problems (1.4) or (1.5) with Σ updated by the Schur complement deflation method (3.1). We denote the leading r sparse PCs obtained by our ADMM as $X_r = (x_1, \cdots, x_r)$. Usually the total variance explained by X_r is given by $\mathbf{Tr}(X_r^\top \Sigma X_r)$. However, because we do not require the loading vectors to be orthogonal to each other when we sequentially solve the SDPs (1.4) or (1.5), these PCs are correlated. Thus, $\mathbf{Tr}(X_r^\top \Sigma X_r)$ will overestimate the total explained variance by $x_1, \cdots, x_r$. To alleviate the overestimated variance, Zou et al. [42] suggested that the explained total variance should be computed using the following procedure, which was called *adjusted variance*:

$$AdjVar(X_r) := \mathbf{Tr}\big(R^2\big),$$

where $M^\top X_r = QR$ is the QR decomposition of $M^\top X_r$. In our numerical experiments, we always report the adjusted variance as the explained variance.

It is also worth noticing that the problems we solve are convex relaxations of the original problem (1.2). Hence, one needs to postprocess the matrix X obtained by solving (1.4) or (1.5) to get the approximate solution to (1.2). To get the solution to the original sparse PCA problem (1.2) from the solution X of the convex SDP problem, we simply perform a rank-one decomposition to X, i.e., $X = xx^\top$. Since X is a sparse matrix, x should be a sparse vector. This postprocessing technique is also used in [9].

For the stopping criteria of ADMM, we consider both the primal and dual residuals as suggested by Boyd et al. in [2]. Note that in our problem, the primal residual at iteration k is measured by

$$r^k := X^k - Y^k$$

and the dual residual at iteration k is measured by

$$s^k := \big(Y^{k-1} - Y^k\big)/\mu.$$

We thus chose to terminate our ADMM when

$$\big\|r^k\big\|_F < \epsilon^p \quad \text{and} \quad \big\|s^k\big\|_F < \epsilon^d, \tag{3.2}$$

where

$$\epsilon^p := p\epsilon_1 + \epsilon_2 \max\{\|X^k\|_F, \|Y^k\|_F\} \quad \text{and} \quad \epsilon^d := p\epsilon_1 + \epsilon_2 \|\Lambda^k\|_F, \tag{3.3}$$

and ϵ_1 and ϵ_2 are tolerance parameters that will be specified later.

4 Numerical Results

In this section, we use our ADMM to solve the SDP formulations (1.4) and (1.5) of sparse PCA on both synthetic and real data sets. We mainly compare the performance of ADMM with DSPCA used in [9] for solving (1.5). We also include a comparison with the ALSPCA method proposed in [23], but we should note that ALSPCA solves a completely different model, which is non-convex and thus the global convergence of ALSPCA is not guaranteed. The MATLAB codes of DSPCA and ALSPCA were downloaded from the authors' websites. Our codes were written in MATLAB. All experiments were run in MATLAB 7.6.0 on a laptop with Intel Core I3 2.30 GHz CPU and 6 GB of RAM.

4.1 Random Examples

We created some random examples to test the speed of ADMM and compared it with DSPCA [9] and ALSPCA [23]. The sample covariance matrix Σ was created by adding some small noise to a sparse rank-one matrix. Specifically, we first created a sparse vector $\hat{x} \in \mathbb{R}^p$ with s nonzeros randomly chosen from the Gaussian distribution $\mathcal{N}(0, 1)$. We then got the sample covariance matrix $\Sigma = \hat{x}\hat{x}^\top + \sigma vv^\top$, where σ denotes the noise level and $v \in \mathbb{R}^p$ is a random vector with entries uniformly drawn from $[0, 1]$. We applied DSPCA and ADMM to find the largest sparse PC of Σ. We report the comparison results in Tables 1 and 2 that correspond to noise levels $\sigma = 0.01$ and $\sigma = 0.1$, respectively. The parameters used for ALSPCA were set as their default settings. When using DSPCA to solve (1.5), we set different ρ's to get solutions with different sparsity levels. Specifically, we tested DSPCA for $\rho = 0.01, 0.1, 1$ in Tables 1 and 2. We set different K's in (1.4) to control the sparsity level when using ADMM to solve it. We set $\epsilon_1 = 10^{-3}$ and $\epsilon_2 = 10^{-4}$ used in (3.3) for the stopping criterion (3.2) of ADMM, and the parameter μ was set as $p/200$. In both Tables 1 and 2, we tested four data sets with dimension p and sparsity s setting as $(p, s) = (100, 10), (100, 20), (200, 10)$ and $(200, 20)$.

We report the cardinality of the largest sparse PC (Card), the percentage of the explained variance (PEV) and the CPU time in Tables 1 and 2. The objective function value $\langle \Sigma, X \rangle$ for both ADMM and DSPCA is also reported. We do not include the objective value for ALSPCA because it solves a different model. From Table 1 we see that, for $\sigma = 0.01$, DSPCA is sensitive to the parameter ρ that controls the sparsity. $\rho = 0.01$ always gave the best results for DSPCA and the explained variance is very close to the standard PCA. $\rho = 0.1$ still provided relatively good solutions for DSPCA in terms of both sparsity and the explained variance. When ρ was increased to 1, the solutions given by DSPCA sometimes had more nonzeros than the desired sparsity level (when $(p, s) = (100, 10)$), and even when the solutions were of the desired

Table 1 Comparisons of ADMM, DSPCA and ALSPCA on random examples with $\sigma = 0.01$

(p, s)	Method	Parameters	Card	PEV (%)	CPU	Obj
(100, 10)	PCA			96.163249		
	DSPCA	$\rho = 0.01$	10	96.159101	6.55	7.345668e+000
		$\rho = 0.10$	10	95.813919	4.83	7.319300e+000
		$\rho = 1.00$	13	87.279890	4.18	6.667379e+000
	ADMM	$K = 7$	19	96.163089	0.07	7.345973e+000
		$K = 6$	9	95.781520	0.06	7.316825e+000
		$K = 5$	7	93.862335	0.06	7.170217e+000
	ALSPCA		9	96.024985	0.07	
(100, 20)	PCA			98.072574		
	DSPCA	$\rho = 0.01$	20	98.068461	6.10	1.489481e+001
		$\rho = 0.10$	20	97.708755	4.85	1.484018e+001
		$\rho = 1.00$	20	85.252763	3.70	1.294834e+001
	ADMM	$K = 13$	20	98.072458	0.10	1.489542e+001
		$K = 12$	20	98.047225	0.11	1.489158e+001
		$K = 11$	19	97.788266	0.15	1.485225e+001
	ALSPCA		18	97.828436	0.07	
(200, 10)	PCA			91.425700		
	DSPCA	$\rho = 0.01$	10	91.421783	30.35	7.345351e+000
		$\rho = 0.10$	10	91.092859	22.11	7.318923e+000
		$\rho = 1.00$	8	82.914799	17.21	6.661851e+000
	ADMM	$K = 9$	10	91.425406	0.26	7.345642e+000
		$K = 8$	10	91.425042	0.26	7.345612e+000
		$K = 7$	9	91.251270	0.26	7.331651e+000
	ALSPCA		9	91.293829	0.12	
(200, 20)	PCA			95.579244		
	DSPCA	$\rho = 0.01$	20	95.575274	28.04	1.489413e+001
		$\rho = 0.10$	20	95.224690	22.16	1.483950e+001
		$\rho = 1.00$	20	83.086984	15.27	1.294800e+001
	ADMM	$K = 14$	20	95.568816	0.58	1.489313e+001
		$K = 13$	20	95.424208	0.56	1.487059e+001
		$K = 12$	19	95.149218	0.59	1.482774e+001
	ALSPCA		18	95.341285	0.10	

sparsity level, the explained variances were affected by a lot (when $(p, s) = (100, 20)$ and $(200, 20)$). For ADMM, different K's were tested for different settings. Results shown in Table 1 indicate that when K is slightly greater than $s/2$, ADMM usually produced good results. For example, for $(p, s) = (100, 20)$, both $K = 13$ and $K = 12$ produced good results in the sense that the cardinality of the largest sparse PC is the same as the desired one, and the explained variance is very close to the standard PCA.

Table 2 Comparisons of ADMM, DSPCA, and ALSPCA on random examples with $\sigma = 0.1$

(p, s)	Method	Parameters	Card	PEV (%)	CPU	Obj
(100, 10)	PCA			71.514735		
	DSPCA	$\rho = 0.01$	46	71.490030	6.50	7.349620e+000
		$\rho = 0.10$	21	71.227570	5.14	7.322638e+000
		$\rho = 1.00$	10	64.918238	2.93	6.673999e+000
	ADMM	$K = 7$	21	71.403999	0.06	7.340776e+000
		$K = 6$	9	71.184328	0.06	7.318192e+000
		$K = 5$	7	69.745358	0.06	7.170257e+000
	ALSPCA		9	71.388004	0.08	
(100, 20)	PCA			83.590727		
	DSPCA	$\rho = 0.01$	62	83.580802	6.91	1.490222e+001
		$\rho = 0.10$	20	83.270360	4.73	1.484687e+001
		$\rho = 1.00$	20	72.747226	4.05	1.297063e+001
	ADMM	$K = 12$	20	83.480007	0.10	1.488425e+001
		$K = 11$	19	83.336977	0.13	1.485875e+001
		$K = 10$	18	82.890834	0.14	1.477920e+001
	ALSPCA		19	83.371441	0.07	
(200, 10)	PCA			51.692525		
	DSPCA	$\rho = 0.01$	10	51.604525	12.49	7.346205e+000
		$\rho = 0.10$	10	51.455062	8.16	7.324928e+000
		$\rho = 1.00$	88	46.945896	7.67	6.683022e+000
	ADMM	$K = 8$	14	51.499128	0.23	7.331202e+000
		$K = 7$	9	51.505655	0.26	7.332131e+000
		$K = 6$	8	50.893478	0.26	7.244984e+000
	ALSPCA		9	51.529656	0.12	
(200, 20)	PCA			68.379142		
	DSPCA	$\rho = 0.01$	20	68.373517	32.02	1.489495e+001
		$\rho = 0.10$	20	68.115662	27.10	1.483878e+001
		$\rho = 1.00$	20	59.539696	21.34	1.297053e+001
	ADMM	$K = 13$	19	68.218241	0.54	1.486113e+001
		$K = 12$	19	68.007900	0.51	1.481530e+001
		$K = 11$	17	67.641121	0.59	1.473540e+001
	ALSPCA		18	68.203072	0.11	

When $K = 11$, ADMM produced a sparser solution, i.e., the cardinality of the largest PC was 19, but the explained variance was only degraded slightly.

From Table 2 we see that, for $\sigma = 0.1$, i.e., when the noise level was larger, DSPCA was more sensitive to the noise compared with their performance when $\sigma = 0.01$. However, we observed that the performance of ADMM when $\sigma = 0.1$ was consistent with its performance when $\sigma = 0.01$, i.e., its performance was not very sensitive to the noise.

From both Tables 1 and 2, we see that ADMM was significantly faster than DSPCA, and comparable to the speed of ALSPCA.

4.2 Text Data Classification

Sparse PCA can also be used to classify the keywords in text data. This application has been studied by Zhang, d'Aspremont and El Ghaoui in [40] and Zhang and El Ghaoui in [41]. In this section, we show that by using our ADMM to solve the sparse PCA problem, we can also classify the keywords from text data very well. The data set we used is a small version of the "20-newsgroups" data,[1] which is also used in [40]. This data set consists of the binary occurrences of 100 specific words across 16242 postings, i.e., the data matrix M is of the size 100×16242 and $M_{ij} = 1$ if the ith word appears at least once in the jth posting and $M_{ij} = 0$ if the ith word does not appear in the jth posting. These words can be approximately divided into different groups such as "computer", "religion" etc. We want to find the words that contribute as much variance as possible and also discover which words are in the same category. By viewing each posting as a sample of the 100 variables, we have 16424 samples of the variables, and thus the sample covariance matrix $\Sigma \in \mathbb{R}^{100\times 100}$. Using standard PCA, it is hard to interpret which words contribute to each of the leading eigenvalues since the loadings are dense. However, sparse PCA can explain as much the variance explained by the standard PCs, and meanwhile interpret well which words contribute together to the corresponding variance. We applied our ADMM to solve (1.4) to find the first three sparse PCs. We set $(\mu, K) = (0.05, 5), (0.01, 3)$ and $(0.01, 4)$, respectively, in the three resulting problems. We set $\epsilon_1 = \epsilon_2 = 10^{-4}$ in the stopping criterion (3.2). The resulting three sparse PCs have eight, 12 and 19 nonzeros, respectively. The total explained variance by these three sparse PCs is 11.64 %, while the variance explained by the largest three PCs by the standard PCA is 19.10 %.

The words corresponding to the first three sparse PCs generated by our ADMM are listed in Table 3. From Table 3 we see that the words in the first sparse PC are approximately in the category "school", the words in the second PC are approximately in the category "religion", and the words in the third sparse PC are approximately in the category "computer". So our ADMM can classify the keywords into appropriate categories very well.

4.3 Senate Voting Data

In this section, we use sparse PCA to analyze the voting records of the 109th US Senate, which was also studied by Zhang, d'Aspremont, and El Ghaoui in [40]. The votes are recorded as 1 for "yes" and -1 for "no". Missing votes are recorded as 0. There are 100 senators (55 Republicans, 44 Democrats and one independent) and 542 bills involved in the data set. However, there are many missing votes in the data set. To obtain a meaningful data matrix, we only choose the bills for which the number of missing votes is at most one. There are only 66 such bills among the 542 bills. So our data matrix M is a 66×100 matrix with entries 1, -1 and 0, and each column of

[1] This data set can be downloaded from http://cs.nyu.edu/~roweis/data.html.

Table 3 Words associated with the first three sparse PCs using ADMM

First PC (eight words)	Second PC (12 words)	Third PC (19 words)
course	bible	computer
email	case	email
fact	christian	files
help	course	ftp
problem	evidence	graphics
question	fact	mac
system	god	number
university	government	phone
	jesus	problem
	number	program
	question	research
	world	science
		software
		state
		system
		university
		version
		windows
		world
Total sparsity: 39, total explained variance: 11.64 %		

M corresponds to one senator's voting. The sample covariance matrix $\Sigma = MM^\top$ in our test is a 66×66 matrix.

To see how standard PCA and sparse PCA perform in classifying the voting records, we implemented the following procedure as suggested in [40]. We used standard PCA to find the largest two PCs (denoted as v_1 and v_2) of Σ. We then projected each column of M onto the subspace spanned by v_1 and v_2, i.e., we found $\bar{\alpha}_i$ and $\bar{\beta}_i$ for each column M_i such that

$$(\bar{\alpha}_i, \bar{\beta}_i) := \arg \min_{(\alpha_i, \beta_i)} \|\alpha_i v_1 + \beta_i v_2 - M_i\|.$$

We then drew each column M_i as a point $(\bar{\alpha}_i, \bar{\beta}_i)$ in the two-dimensional subspace spanned by v_1 and v_2. The left figure in Fig. 1 shows the 100 points. We see from this figure that senators are separated very well by partisanship. However, it is hard to interpret which bills are responsible to the explained variance, because all the bills are involved in the PCs. By using sparse PCA, we can interpret the explained variance by just a few bills. We applied our ADMM to find the first two sparse PCs (denoted as s_1 and s_2) of Σ. We set $(\mu, K) = (0.05, 5)$ for both problems and $\epsilon_1 = \epsilon_2 = 10^{-4}$ in the stopping criterion (3.2).

The resulting two sparse PCs s_1 and s_2 produced by our ADMM have eight and six nonzeros, respectively. We projected each column of M onto the subspace spanned by these two sparse PCs. The right figure in Fig. 1 shows the 100 projections onto the

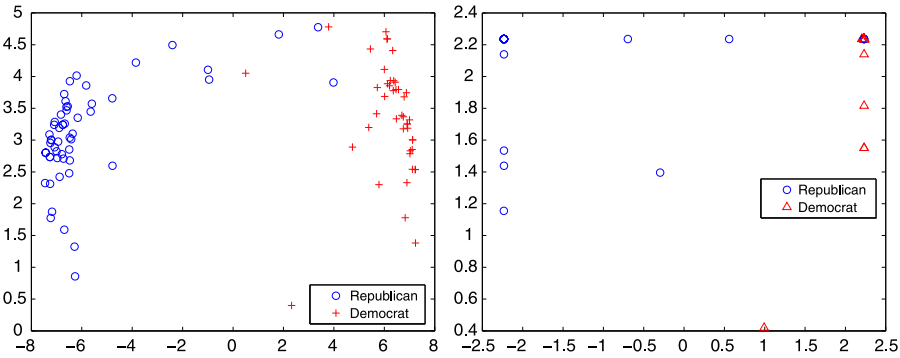

Fig. 1 Projection of the senate voting records onto the subspace spanned by the top 2 principal components: *Left*: standard PCA; *Right*: sparse PCA

Table 4 Bills involved in the first two PCs by ADMM

Bills in the first sparse PC
Budget, Spending, and Taxes_Education Funding Amendment_3804
Budget, Spending, and Taxes_Reinstate Pay-As-You-Go through 2011 Amendment_3806
Energy Issues_LIHEAP Funding Amendment_3808
Abortion Issues_Unintended Pregnancy Amendment_3489
Budget, Spending, and Taxes_Budget FY2006 Appropriations Resolution_3488
Budget, Spending, and Taxes_Budget Reconciliation bill_3665
Budget, Spending, and Taxes_Budget Reconciliation bill_3789
Budget, Spending, and Taxes_Education Amendment_3490
Health Issues_Medicaid Amendment_3496
Bills in the second sparse PC
Appropriations_Agriculture, Rural Development, FDA Appropriations Act_3677
Appropriations_Emergency Supplemental Appropriations Act, 2005_3515
Appropriations_Emergency Supplemental Appropriations Act, 2006_3845
Appropriations_Interior Department FY 2006 Appropriations Bill_3595
Executive Branch_John Negroponte, Director of National Intelligence_3505

subspace spanned by the sparse PCs s_1 and s_2. We see from this figure that the senators are still separated well by partisanship. Now since only a few bills are involved in the two sparse PCs, we can interpret which bills are responsible most for the classification. The bills involved in the first two PCs are listed in Table 4. From Table 4 we see that the most controversial issues between Republicans and Democrats are topics such as "Budget" and "Appropriations". Other controversial issues involve topics like "Energy", "Abortion" and "Health".

5 Conclusion

In this paper, we proposed alternating direction method of multipliers to solve an SDP relaxation of the sparse PCA problem. Our method incorporated a variable-splitting technique to separate the ℓ_1 norm constraint, which controls the sparsity of the solution, and the positive-semidefiniteness constraint. This method resulted in two relatively simple subproblems that have closed-form solutions in each iteration. Global convergence results were established for the proposed method. Numerical results on both synthetic data and real data from classification of text data and senate voting records demonstrated the efficacy of our method. Compared with Nesterov's first-order method DSPCA for sparse PCA studied in [9], our ADMM method solves the primal problems directly and guarantees sparse solutions. Numerical results also indicate that ADMM is much faster than DSPCA.

Acknowledgements The author is grateful to Alexandre d'Aspremont for discussions on using DSPCA. The author thanks Stephen J. Wright and Lingzhou Xue for reading an earlier version of the manuscript and for helpful discussions. The author also thanks the anonymous reviewers for constructive suggestions that greatly improved the presentation of this paper.

Appendix

In this section, we prove that the sequence (X^k, Y^k, Λ^k) produced by the alternating direction method of multipliers (2.7) (i.e., Algorithm 2) converges to (X^*, Y^*, Λ^*), where (X^*, Y^*) is an optimal solution to (2.4) and Λ^* is the corresponding optimal dual variable. Although the proof of global convergence results of ADMM has been studied extensively in the literature (see e.g., [14, 20]), we here give a very simple proof of the convergence of our ADMM that utilizes the special structures of the sparse PCA problem. We only prove the case when $h(Y) = I_{\mathcal{B}}(Y)$ and leave the case when $h(Y) = \rho\|Y\|_1$ to the readers since their proofs are almost identical.

Before we give the main theorem about the global convergence of (2.7) (Algorithm 2), we need the following lemma.

Lemma A.1 *Assume that (X^*, Y^*) is an optimal solution of (2.4) and Λ^* is the corresponding optimal dual variable associated with the equality constraint $X = Y$. Then the sequence (X^k, Y^k, Λ^k) produced by (2.7) satisfies*

$$\left\|U^k - U^*\right\|_G^2 - \left\|U^{k+1} - U^*\right\|_G^2 \geqslant \left\|U^k - U^{k+1}\right\|_G^2, \tag{A.1}$$

where

$$U^* = \begin{pmatrix} \Lambda^* \\ Y^* \end{pmatrix}, \quad U^k = \begin{pmatrix} \Lambda^k \\ Y^k \end{pmatrix} \quad \text{and} \quad G = \begin{pmatrix} \mu I & 0 \\ 0 & \frac{1}{\mu} I \end{pmatrix},$$

and the norm $\|\cdot\|_G^2$ is defined as $\|U\|_G^2 = \langle U, GU\rangle$ and the corresponding inner product $\langle\cdot,\cdot\rangle_G$ is defined as $\langle U, V\rangle_G = \langle U, GV\rangle$.

Proof Since (X^*, Y^*, Λ^*) is optimal to (2.4), it follows from the KKT conditions that the following hold:

$$0 \in -\Sigma + \partial I_{\mathcal{C}}(X^*) - \Lambda^*, \tag{A.2}$$

$$0 \in \partial I_{\mathcal{B}}(Y^*) + \Lambda^*, \tag{A.3}$$

and

$$X^* = Y^* \in \mathcal{C} \cap \mathcal{B}. \tag{A.4}$$

By using Property 2, (A.2) and (A.3) can be, respectively, reduced to

$$\langle \Sigma + \Lambda^*, X - X^* \rangle \geqslant 0, \quad \forall X \in \mathcal{C}, \tag{A.5}$$

and

$$\langle -\Lambda^*, Y - Y^* \rangle \geqslant 0, \quad \forall Y \in \mathcal{B}. \tag{A.6}$$

Note that the optimality conditions for the first subproblem (i.e., the subproblem with respect to X) in (2.7) are given by $X^{k+1} \in \mathcal{C}$ and

$$0 \in -\Sigma + \partial I_{\mathcal{C}}(X^{k+1}) - \Lambda^k + \frac{1}{\mu}(X^{k+1} - Y^k). \tag{A.7}$$

By using Property 2 and the updating formula for Λ^k in (2.7), i.e.,

$$\Lambda^{k+1} = \Lambda^k - \frac{1}{\mu}(X^{k+1} - Y^{k+1}), \tag{A.8}$$

(A.7) can be rewritten as

$$\left\langle \Sigma + \Lambda^{k+1} + \frac{1}{\mu}(Y^k - Y^{k+1}), X - X^{k+1} \right\rangle \geqslant 0, \quad \forall X \in \mathcal{C}. \tag{A.9}$$

Letting $X = X^{k+1}$ in (A.5) and $X = X^*$ in (A.9), and summing the two resulting inequalities, we get

$$\left\langle \Lambda^{k+1} - \Lambda^* + \frac{1}{\mu}(Y^k - Y^{k+1}), X^* - X^{k+1} \right\rangle \geqslant 0. \tag{A.10}$$

The optimality conditions for the second subproblem (i.e., the subproblem with respect to Y) in (2.7) are given by $Y^{k+1} \in \mathcal{B}$ and

$$0 \in \partial I_{\mathcal{B}}(Y^{k+1}) + \Lambda^k + \frac{1}{\mu}(Y^{k+1} - X^{k+1}). \tag{A.11}$$

By using Property 2 and (A.8), (A.11) can be rewritten as

$$\langle -\Lambda^{k+1}, Y - Y^{k+1} \rangle \geqslant 0, \quad \forall Y \in \mathcal{B}. \tag{A.12}$$

Letting $Y = Y^{k+1}$ in (A.6) and $Y = Y^*$ in (A.12), and summing the two resulting inequalities, we obtain

$$\left\langle \Lambda^* - \Lambda^{k+1}, Y^* - Y^{k+1} \right\rangle \leqslant 0. \tag{A.13}$$

Summing (A.10) and (A.13), and using the facts that $X^* = Y^*$ and $X^{k+1} = \mu(\Lambda^k - \Lambda^{k+1}) + Y^{k+1}$, we obtain

$$\mu\left\langle \Lambda^k - \Lambda^{k+1}, \Lambda^{k+1} - \Lambda^* \right\rangle + \frac{1}{\mu}\left\langle Y^k - Y^{k+1}, Y^{k+1} - Y^* \right\rangle \geqslant -\left\langle Y^k - Y^{k+1}, \Lambda^k - \Lambda^{k+1} \right\rangle. \tag{A.14}$$

Rearranging the left hand side of (A.14) by using $\Lambda^{k+1} - \Lambda^* = (\Lambda^{k+1} - \Lambda^k) + (\Lambda^k - \Lambda^*)$ and $Y^{k+1} - Y^* = (Y^{k+1} - Y^k) + (Y^k - Y^*)$, we get

$$\begin{aligned} &\mu\left\langle \Lambda^k - \Lambda^*, \Lambda^k - \Lambda^{k+1} \right\rangle + \frac{1}{\mu}\left\langle Y^k - Y^*, Y^k - Y^{k+1} \right\rangle \\ &\quad \geqslant \mu\left\| \Lambda^k - \Lambda^{k+1} \right\|^2 + \frac{1}{\mu}\left\| Y^k - Y^{k+1} \right\|^2 - \left\langle \Lambda^{k+1} - \Lambda^k, Y^{k+1} - Y^k \right\rangle. \end{aligned} \tag{A.15}$$

Using the notation of U^k, U^* and G, (A.15) can be rewritten as

$$\left\langle U^k - U^*, U^k - U^{k+1} \right\rangle_G \geqslant \left\| U^k - U^{k+1} \right\|_G^2 - \left\langle \Lambda^k - \Lambda^{k+1}, Y^k - Y^{k+1} \right\rangle. \tag{A.16}$$

Combining (A.16) with the identity

$$\left\| U^{k+1} - U^* \right\|_G^2 = \left\| U^{k+1} - U^k \right\|_G^2 - 2\left\langle U^k - U^{k+1}, U^k - U^* \right\rangle_G + \left\| U^k - U^* \right\|_G^2,$$

we get

$$\begin{aligned} &\left\| U^k - U^* \right\|_G^2 - \left\| U^{k+1} - U^* \right\|_G^2 \\ &\quad = 2\left\langle U^k - U^{k+1}, U^k - U^* \right\rangle - \left\| U^{k+1} - U^k \right\|_G^2 \\ &\quad \geqslant 2\left\| U^k - U^{k+1} \right\|_G^2 - 2\left\langle \Lambda^k - \Lambda^{k+1}, Y^k - Y^{k+1} \right\rangle - \left\| U^{k+1} - U^k \right\|_G^2 \\ &\quad = \left\| U^k - U^{k+1} \right\|_G^2 - 2\left\langle \Lambda^k - \Lambda^{k+1}, Y^k - Y^{k+1} \right\rangle. \end{aligned} \tag{A.17}$$

Now, using (A.12) for k instead of $k+1$ and letting $Y = Y^{k+1}$, we get

$$\left\langle -\Lambda^k, Y^{k+1} - Y^k \right\rangle \leqslant 0. \tag{A.18}$$

Letting $Y = Y^k$ in (A.12) and adding it to (A.18) yields

$$\left\langle \Lambda^k - \Lambda^{k+1}, Y^k - Y^{k+1} \right\rangle \leqslant 0. \tag{A.19}$$

By substituting (A.19) into (A.17) we get the desired result (A.1). □

We are now ready to give the main convergence result of (2.7) (Algorithm 2).

Theorem A.2 *The sequence* $\{(X^k, Y^k, \Lambda^k)\}$ *produced by* (2.7) (*Algorithm* 2) *from any starting point converges to an optimal solution to Problem* (2.4).

Proof From Lemma A.1 we can easily get

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

It follows from (i) that $\Lambda^k - \Lambda^{k+1} \to 0$ and $Y^k - Y^{k+1} \to 0$. Then (A.8) implies that $X^k - X^{k+1} \to 0$ and $X^k - Y^k \to 0$. From (ii) we obtain the result that U^k has a subsequence $\{U^{k_j}\}$ that converges to $\hat{U} = (\hat{\Lambda}, \hat{Y})$, i.e., $\Lambda^{k_j} \to \hat{\Lambda}$ and $Y^{k_j} \to \hat{Y}$. From $X^k - Y^k \to 0$ we also get $X^{k_j} \to \hat{X} := \hat{Y}$. Therefore, $(\hat{X}, \hat{Y}, \hat{\Lambda})$ is a limit point of $\{(X^k, Y^k, \Lambda^k)\}$.

Note that by using (A.8), (A.7) can be rewritten as

$$0 \in -\Sigma + \partial I_{\mathcal{C}}\big(X^{k+1}\big) - \Lambda^{k+1} + \frac{1}{\mu}\big(Y^{k+1} - Y^k\big), \tag{A.20}$$

which implies that

$$0 \in -\Sigma + \partial I_{\mathcal{C}}(\hat{X}) - \hat{\Lambda}. \tag{A.21}$$

Note also that (A.11) implies that

$$0 \in \partial I_{\mathcal{B}}(\hat{Y}) + \hat{\Lambda}. \tag{A.22}$$

Moreover, it follows from $X^k \in \mathcal{C}$ and $Y^k \in \mathcal{B}$ that

$$\hat{X} \in \mathcal{C} \quad \text{and} \quad \hat{Y} \in \mathcal{B}. \tag{A.23}$$

(A.21), (A.22), (A.23) together with $\hat{X} = \hat{Y}$ imply that $(\hat{X}, \hat{Y}, \hat{\Lambda})$ satisfies the KKT conditions for (2.4) and thus is an optimal solution to (2.4).

To complete the proof, we need to show that the limit point is unique. 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 (2.4). Thus, U^* in (A.1) can be replaced by $\hat{U}_1 := (\hat{R}_1, \hat{W}_1, \hat{\Lambda}_1)$ and $\hat{U}_2 := (\hat{R}_2, \hat{W}_2, \hat{\Lambda}_2)$. This results in

$$\big\|U^{k+1} - \hat{U}_i\big\|_G^2 \leqslant \big\|U^k - \hat{U}_i\big\|_G^2, \quad i = 1, 2,$$

and we thus get the existence of the limits

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

Now using the identity

$$\big\|U^k - \hat{U}_1\big\|_G^2 - \big\|U^k - \hat{U}_2\big\|_G^2 = -2\big\langle U^k, \hat{U}_1 - \hat{U}_2\big\rangle_G + \|\hat{U}_1\|_G^2 - \|\hat{U}_2\|_G^2$$

and passing the limit we get

$$\eta_1^2 - \eta_2^2 = -2\langle \hat{U}_1, \hat{U}_1 - \hat{U}_2\rangle_G + \|\hat{U}_1\|_G^2 - \|\hat{U}_2\|_G^2 = -\|\hat{U}_1 - \hat{U}_2\|_G^2$$

and

$$\eta_1^2 - \eta_2^2 = -2\langle \hat{U}_2, \hat{U}_1 - \hat{U}_2 \rangle_G + \|\hat{U}_1\|_G^2 - \|\hat{U}_2\|_G^2 = \|\hat{U}_1 - \hat{U}_2\|_G^2.$$

Thus we must have $\|\hat{U}_1 - \hat{U}_2\|_G^2 = 0$ and hence the limit point of $\{(X^k, Y^k, \Lambda^k)\}$ is unique. □

References

[1] Alizadeh, F.: Interior point methods in semidefinite programming with applications to combinatorial optimization. SIAM J. Optim. **5**, 13–51 (1993)

[2] Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Found. Trends Mach. Learn. **3**, 1–122 (2011)

[3] Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, Cambridge (2004)

[4] Brucker, P.: An $\mathcal{O}(n)$ algorithm for quadratic knapsack problems. Oper. Res. Lett. **3**, 163–166 (1984)

[5] Candès, E.J., Romberg, J., Tao, T.: Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. IEEE Trans. Inf. Theory **52**, 489–509 (2006)

[6] Combettes, P.L., Pesquet, J.-C.: A Douglas-Rachford splitting approach to nonsmooth convex variational signal recovery. IEEE J. Sel. Top. Signal Process. **1**, 564–574 (2007)

[7] Combettes, P.L., Wajs, V.R.: Signal recovery by proximal forward–backward splitting. SIAM J. Multiscale Model. Simul. **4**, 1168–1200 (2005)

[8] d'Aspremont, A., Bach, F., El Ghaoui, L.: Optimal solutions for sparse principal component analysis. J. Mach. Learn. Res. **9**, 1269–1294 (2008)

[9] d'Aspremont, A., El Ghaoui, L., Jordan, M.I., Lanckriet, G.R.G.: A direct formulation for sparse pca using semidefinite programming. SIAM Rev. **49**, 434–448 (2007)

[10] Donoho, D.: Compressed sensing. IEEE Trans. Inf. Theory **52**, 1289–1306 (2006)

[11] Douglas, J., Rachford, H.H.: On the numerical solution of the heat conduction problem in 2 and 3 space variables. Trans. Am. Math. Soc. **82**, 421–439 (1956)

[12] Duchi, J., Shalev-Shwartz, S., Singer, Y., Chandra, T.: Efficient projections onto the l1-ball for learning in high dimensions. In: ICML (2008)

[13] Eckstein, J.: Splitting methods for monotone operators with applications to parallel optimization. PhD thesis, Massachusetts Institute of Technology (1989)

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

[15] Gabay, D.: Applications of the method of multipliers to variational inequalities. In: Fortin, M., Glowinski, R. (eds.) Augmented Lagrangian Methods: Applications to the Solution of Boundary Value Problems. North-Holland, Amsterdam (1983)

[16] Glowinski, R., Le Tallec, P.: Augmented Lagrangian and Operator-Splitting Methods in Nonlinear Mechanics. SIAM, Philadelphia (1989)

[17] Goldfarb, D., Ma, S.: Fast multiple splitting algorithms for convex optimization. SIAM J. Optim. **22**, 533–556 (2012)

[18] Goldfarb, D., Ma, S., Scheinberg, K.: Fast alternating linearization methods for minimizing the sum of two convex functions. Math. Program. (2012). doi:10.1007/s10107-012-0530-2

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

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

[21] Journee, M., Nesterov, Yu., Richtarik, P., Sepulchre, R.: Generalized power method for sparse principal component analysis. J. Mach. Learn. Res. **11**, 517–553 (2010)

[22] Lions, P.L., Mercier, B.: Splitting algorithms for the sum of two nonlinear operators. SIAM J. Numer. Anal. **16**, 964–979 (1979)

[23] Lu, Z., Zhang, Y.: An augmented Lagrangian approach for sparse principal component analysis. Math. Program. **135**, 149–193 (2012)

[24] Luss, R., Teboulle, M.: Conditional gradient algorithms for rank-one matrix approximations with a sparsity constraint. SIAM Rev.

[25] Mackey, L.: Deflation methods for sparse PCA. In: Advances in Neural Information Processing Systems (NIPS) (2008)
[26] Malick, J., Povh, J., Rendl, F., Wiegele, A.: Regularization methods for semidefinite programming. SIAM J. Optim. **20**, 336–356 (2009)
[27] Nesterov, Y.E.: A method for unconstrained convex minimization problem with the rate of convergence $\mathcal{O}(1/k^2)$. Dokl. Akad. Nauk SSSR **269**, 543–547 (1983)
[28] Nesterov, Y.E.: Smooth minimization for non-smooth functions. Math. Program., Ser. A **103**, 127–152 (2005)
[29] Pardalos, P.M., Kovoor, N.: An algorithm for a singly constrained class of quadratic programs subject to upper and lower bounds. Math. Program. **46**, 321–328 (1990)
[30] Peaceman, D.H., Rachford, H.H.: The numerical solution of parabolic elliptic differential equations. SIAM J. Appl. Math. **3**, 28–41 (1955)
[31] Shalev-Shwartz, S., Singer, Y.: Efficient learning of label ranking by soft projections onto polyhedra. J. Mach. Learn. Res. **7**, 1567–1599 (2006)
[32] Saad, Y.: Projection and deflation methods for partial pole assignment in linear state feedback. IEEE Trans. Autom. Control **33**, 290–297 (1998)
[33] Scheinberg, K., Ma, S., Goldfarb, D.: Sparse inverse covariance selection via alternating linearization methods. In: NIPS (2010)
[34] Todd, M.J.: Semidefinite optimization. Acta Numer. **10**, 515–560 (2001)
[35] van den Berg, E., Friedlander, M.P.: Probing the Pareto frontier for basis pursuit solutions. SIAM J. Sci. Comput. **31**, 890–912 (2008)
[36] Wang, Y., Yang, J., Yin, W., Zhang, Y.: A new alternating minimization algorithm for total variation image reconstruction. SIAM J. Imaging Sci. **1**, 248–272 (2008)
[37] Wen, Z., Goldfarb, D., Yin, W.: Alternating direction augmented Lagrangian methods for semidefinite programming. Math. Program. Comput. **2**, 203–230 (2010)
[38] Yang, J., Zhang, Y.: Alternating direction algorithms for ℓ_1 problems in compressive sensing. SIAM J. Sci. Comput. **33**, 250–278 (2011)
[39] Yuan, X.: Alternating direction methods for sparse covariance selection. J. Sci. Comput. **51**, 261–273 (2012)
[40] Zhang, Y., d'Aspremont, A., El Ghaoui, L.: Sparse PCA: Convex relaxations, algorithms and applications. In: Anjos, M., Lasserre, J.B. (eds.) Handbook on Semidefinite, Cone and Polynomial Optimization (2011)
[41] Zhang, Y., El Ghaoui, L.: Large-scale sparse principal component analysis with application to text data. In: NIPS (2011)
[42] Zou, H., Hastie, T., Tibshirani, R.: Sparse principle component analysis. J. Comput. Graph. Stat. **15**, 265–286 (2006)