Journal of Inequalities and Applications
a SpringerOpen Journal

RESEARCH Open Access

A customized proximal point algorithm for stable principal component pursuit with nonnegative constraint

Kaizhan Huai*, Mingfang Ni, Feng Ma and Zhanke Yu

*Correspondence:
huaikaizhan@163.com
College of Communications Engineering, PLA University of Science and Technology, Nanjing, 210007, China

Abstract
The stable principal component pursuit (SPCP) problem represents a large class of mathematical models appearing in sparse optimization-related applications such as image restoration, web data ranking. In this paper, we focus on designing a new primal-dual algorithm for the SPCP problem with nonnegative constraint. Our method is based on the framework of proximal point algorithm. By taking full exploitation to the special structure of the SPCP problem, the method enjoys the advantage of being easily implementable. Global convergence result is established for the proposed method. Preliminary numerical results demonstrate that the method is efficient.

Keywords: proximal point method; customized; stable principal component pursuit; primal-dual algorithm

1 Introduction

With the development of information technology, the subject of high-dimensional data becomes more and more popular in science and engineering applications such as image and video processing, web documents analysis and bioinformatics data processing. An intensive research attention has been devoted recently to analyzing, processing and exacting useful information from the high-dimensional data efficiently and accurately. The classical principal component analysis (PCA) is the most widely used tool for high-dimensional data analysis, and it plays a fundamental role in dimensionality reduction. PCA computes the singular value decomposition (SVD) of a matrix to obtain a low-dimensional approximation to high-dimensional data in the ℓ_2 sense [1]. However, PCA usually breaks down when the given data is corrupted by gross errors. In other words, the classical PCA is not robust to gross errors or outliers. To overcome this issue, many methods have been proposed. In [2], a new model called principal component pursuit (PCP) was proposed by Candès and Wright under weak assumptions. It is assumed that the matrix $M \in \mathbb{R}^{m\times n}$ is of the form $M = L + S$, where L is the underlying low-rank matrix representing the principle components and S is a sparse matrix with its most entries being zero. To recover L

and S, PCP requires to solve the following convex optimization problem:

$$\begin{aligned} \min \quad & \|L\|_* + \rho\|S\|_1 \\ \text{s.t.}\ & L + S = M, \\ & L, S \in \mathbb{R}^{m\times n}, \end{aligned} \tag{1.1}$$

where $\|L\|_*$ denotes the nuclear norm of L, which is equal to the sum of its singular values, $\|S\|_1 = \sum_{i,j}|S_{i,j}|$ is the ℓ_1 norm of S, and ρ is a parameter balancing the low-rank and sparsity.

In [3], it was shown that the recovery is still feasible even when the data matrix M is corrupted with a dense error matrix Z such that $\|Z\|_F \leq \delta$. Indeed, this can be accomplished by solving the following stable principal component pursuit (SPCP) problem:

$$\begin{aligned} \min \quad & \|L\|_* + \rho\|S\|_1 \\ \text{s.t.}\ & L + S + Z = M, \\ & \|Z\|_F \leq \sigma, \\ & L, S, Z \in \mathbb{R}^{m\times n}, \end{aligned} \tag{1.2}$$

where the matrix Z is the noise, $\|Z\|_F$ denotes the Frobenius norm of Z, and $\sigma > 0$ is the noise level. Note that (1.1) is a special case of the SPCP problem (1.2) with $\sigma = 0$. In practical applications such as background exacting from face recognition, video denoising and surveillance video, the low-rank matrix L always represents an image. Therefore, adding a nonnegative constraint $L \geq 0$ to (1.2) makes sense. This results in the following SPCP problem with nonnegative constraint:

$$\begin{aligned} \min \quad & \|L\|_* + \rho\|S\|_1 + \mathcal{I}(\|Z\|_F \leq \sigma) + \mathcal{I}(L \geq 0) \\ \text{s.t.}\ & L + S + Z = M, \\ & L, S, Z \in \mathbb{R}^{m\times n}, \end{aligned} \tag{1.3}$$

where $\mathcal{I}(\cdot)$ is an indicator function [4].

Recently, many algorithms using only first-order information for solving the SPCP problem (1.2) have been proposed. Aybat and Iyengar proposed a first-order augmented Lagrangian algorithm (FALC) which was the first algorithm with a known complexity bound that solves the SPCP problem in [5]. Tao and Yuan developed the alternating splitting augmented Lagrangian method (ASALM) and its variant (VASALM) for solving (1.2) in [6]. Aybat *et al.* advanced a new first-order algorithm NSA based on partial variable splitting in [7]. Nevertheless, how to solve (1.3) has not caused enough attention. We can only find that Ma proposed an alternating proximal gradient method (APGM) which was based on the framework of alternating direction method of multipliers for solving (1.3) in [4]. In this paper, we propose a customized proximal point algorithm with a special proximal regularization parameter to solve the SPCP problem. Note that (1.3) is well-structured in the sense that the separable structure emerges in both the objective function and the constraints. A natural idea is to develop a customized algorithm to take advantage of the favorable structure of (1.3). This is the main motivation of our paper.

The rest of this paper is organized as follows. In Section 2, we give some useful preliminaries. In Section 3, we present the customized PPA for solving (1.3) and the convergence analysis is shown in Section 4. In Section 5, we compare our algorithm with APGM to illustrate the efficiency by performing numerical experiments. Finally, some conclusions are drawn in Section 6.

2 Preliminaries

In order to facilitate the analysis, we consider the following separable convex optimization problem with linear constraint instead of (1.3):

$$\begin{aligned} \min \quad & f(x) + g(y) \\ \text{s.t. } & Ax + By = b, \\ & x \in \mathcal{X}, \qquad y \in \mathcal{Y}, \end{aligned} \tag{2.1}$$

where $A \in \mathbb{R}^{m\times n}$, $B \in \mathbb{R}^{m\times p}$, $b \in \mathbb{R}^m$, $\mathcal{X} \subseteq \mathbb{R}^n$ and $\mathcal{Y} \subseteq \mathbb{R}^p$ are convex sets, and $f(x) : \mathbb{R}^n \to \mathbb{R}$ and $g(y) : \mathbb{R}^p \to \mathbb{R}$ are both convex but not necessarily smooth functions [8]. Throughout, the solution set of (2.1) is assumed to be nonempty. Furthermore, we assume that $f(x)$ and $g(y)$ are 'simple' which means that their proximal operators have a closed-form representation or they can be efficiently solved up to a high precision [9]. The proximal operator of the function $\varphi(x) : \mathbb{R}^n \to \mathbb{R}$ is defined as

$$\operatorname{Prox}(\varphi, \xi, a) = \operatorname{argmin}\left\{ \varphi(x) + \frac{1}{2\xi}\|x - a\|^2 \,\middle|\, x \in \mathcal{X} \right\} \tag{2.2}$$

for any given $a \in \mathbb{R}^n$ and $\xi > 0$ [10]. The nuclear norm of L and the ℓ_1 norm of S in (1.3) both are simple functions. Under the assumption, we will show that our algorithm for solving (1.3) can result in easy proximal subproblems.

We signify the subdifferential of the convex function $f(x)$ by $\partial f(x)$,

$$\partial f(x) := \left\{ d \in \mathbb{R}^n \mid f(z) - f(x) \geq d^T(z - x), \forall z \in \mathbb{R}^n \right\}, \tag{2.3}$$

and each $d \in \partial f(x)$ is called a subgradient of $f(x)$ [11]. Let $\theta(x) \in \partial f(x)$, then we can have

$$f(z) - f(x) \geq (z - x)^T \theta(x), \tag{2.4a}$$

$$f(x) - f(z) \geq (x - z)^T \theta(z). \tag{2.4b}$$

Merging (2.4a) and (2.4b), we can easily get

$$(x - z)^T \bigl(\theta(x) - \theta(z)\bigr) \geq 0, \quad \forall x, z \in \mathbb{R}^n, \tag{2.5}$$

which implies that the mapping $\theta(\cdot)$ is monotone.

Now, we show that (2.1) can be characterized by a variational inequality (VI) framework. Let $\lambda \in \mathbb{R}^m$ be the Lagrangian multiplier associated with the linear constraint in (2.1), then the Lagrangian function of (2.1) is

$$\mathcal{L}(x, y; \lambda) = f(x) + g(y) - \lambda^T(Ax + By - b). \tag{2.6}$$

By deriving the optimality conditions of (2.1), we can easily find that solving (2.1) is equivalent to finding a pair of $(x^*, y^*; \lambda^*)$ which satisfies

$$\begin{cases} x^* \in \mathcal{X}, \quad (x - x^*)^T(\theta(x^*) - A^T\lambda^*) \geq 0, \quad \forall x \in \mathcal{X}, \\ y^* \in \mathcal{Y}, \quad (y - y^*)^T(\gamma(y^*) - B^T\lambda^*) \geq 0, \quad \forall y \in \mathcal{Y}, \\ Ax^* + By^* - b = 0, \end{cases} \tag{2.7}$$

where $\theta(x^*) \in \partial f(x^*)$ and $\gamma(y^*) \in \partial g(y^*)$. By denoting

$$u = \begin{pmatrix} x \\ y \end{pmatrix}, \qquad w = \begin{pmatrix} x \\ y \\ \lambda \end{pmatrix}, \qquad h(u) = f(x) + g(y), \qquad F(w) = \begin{pmatrix} -A^T\lambda \\ -B^T\lambda \\ Ax + By - b \end{pmatrix}, \tag{2.8a}$$

and

$$\Omega = \mathcal{X} \times \mathcal{Y} \times \mathbb{R}^m, \tag{2.8b}$$

problem (2.7) can be rewritten as the variational inequality reformulation

$$w^* \in \Omega, \quad h(u) - h(u^*) + (w - w^*)^T F(w^*) \geq 0, \quad \forall w \in \Omega. \tag{2.9}$$

Obviously, the mapping $F(w)$ defined in (2.8a) is monotone. The solution set of (2.8a)-(2.9), denoted by Ω^* is nonempty under the assumption that the solution set of (2.1) is not empty.

3 The new algorithm

In this section, we will present our new algorithm to solve VI (2.9). However, at the beginning, we first review the classical PPA.

After the PPA was proposed firstly by Martinet in [12] and further developed by Rockafellar in [13], it plays a vital role in optimization area. Given the iterate w^k, the classical PPA generates the new iterate $w^{k+1} \in \Omega$ via the following procedure:

$$h(u) - h(u^{k+1}) + (w - w^{k+1})^T (F(w^{k+1}) + G(w^{k+1} - w^k)) \geq 0, \quad \forall u \in \Omega, \tag{3.1}$$

where the metric proximal parameter $G \in \mathbb{R}^{n\times n}$ is required to be a positive definite matrix. A popular choice of G is $G = \beta I$, where $\beta > 0$ and I is the identity matrix [14]. Here, we are ready to present our new algorithm for solving (2.1).

Algorithm 1 (The main algorithm for (2.1))

Let $r > 0$ and $s > \frac{1}{r}\|B^TB\|$, take $(x^0, y^0; \lambda^0) \in \mathcal{X} \times \mathcal{Y} \times \mathbb{R}^m$ as the initial point.

Step 1. Update y, x and λ:

$$y^{k+1} = \operatorname{argmin}\left\{ g(y) + \frac{r}{2}\left\| y - y^k - \frac{1}{r}B^T\lambda^k \right\|^2 \,\middle|\, y \in \mathcal{Y} \right\},$$

$$x^{k+1} = \operatorname{argmin}\left\{ f(x) - \left(\lambda^k - \frac{1}{s}(Ax^k + B(2y^{k+1} - y^k) - b) \right)^T Ax \right.$$

$$+\frac{1}{2s}\left\|A\left(x-x^k\right)\right\|^2+\frac{1}{2}\left\|x-x^k\right\|^2 \,\Big|\, x\in\mathcal{X}\Big\},$$
$$\lambda^{k+1}=\lambda^k-\frac{1}{s}\left(Ax^{k+1}+B\left(2y^{k+1}-y^k\right)-b\right).$$

Step 2. If the termination criterion is met, stop the algorithm; otherwise, go to Step 1.

The new customized PPA described above is known as alternating direction method of multipliers (ADMM) with two blocks [15]. Its global convergence result has been proven in many literature works. However, there are three variables in (1.3). If we apply the customized PPA to the SPCP problem directly, the convergence of the algorithm cannot be guaranteed [16]. Moreover, the proximal mapping of $\|L\|_* + \mathcal{I}(L \geq 0)$ is difficult to compute [4]. By introducing a new auxiliary parameter K, we can group L and S as one big block $[L;S]$, and group Z and K as another big block $[Z;K]$. Then (1.3) can be rewritten as a similar form of (2.1):

$$\begin{aligned}&\min_{L,S,Z,K}\ \|L\|_*+\rho\|S\|_1+\mathcal{I}(\|Z\|_F\leq\sigma)+\mathcal{I}(K\geq 0)\\&\text{s.t.}\quad \begin{pmatrix}I&0\\0&I\end{pmatrix}\begin{pmatrix}Z\\K\end{pmatrix}+\begin{pmatrix}I&I\\-I&0\end{pmatrix}\begin{pmatrix}L\\S\end{pmatrix}=\begin{pmatrix}M\\0\end{pmatrix}.\end{aligned}\tag{3.2}$$

Then we can solve (1.3) or (3.2) by applying the new customized PPA as follows:

$$L^{k+1}=\operatorname{argmin}\|L\|_*+\frac{r}{2}\left\|L-L^k-\frac{1}{r}\left(\Lambda_1^k-\Lambda_2^k\right)\right\|_F^2,\tag{3.3a}$$

$$S^{k+1}=\operatorname{argmin}\rho\|S\|_1+\frac{r}{2}\left\|S-S^k-\frac{1}{r}\Lambda_1^k\right\|_F^2,\tag{3.3b}$$

$$\begin{aligned}Z^{k+1}&=\operatorname{argmin}\mathcal{I}\left(\|Z\|_F\leq\sigma\right)\\&\quad+\frac{1+s}{2s}\left\|Z-Z^k-\frac{1}{1+s}\left(s\Lambda_1^k-\left(2L^{k+1}-L^k+2S^{k+1}-S^k+Z^k-M\right)\right)\right\|_F^2,\end{aligned}\tag{3.3c}$$

$$\begin{aligned}K^{k+1}&=\operatorname{argmin}\mathcal{I}(K\geq 0)\\&\quad+\frac{1+s}{2s}\left\|K-K^k-\frac{1}{1+s}\left(s\Lambda_2^k-\left(L^k-2L^{k+1}+K^k+M\right)\right)\right\|_F^2,\end{aligned}\tag{3.3d}$$

$$\Lambda_1^{k+1}=\Lambda_1^k-\frac{1}{s}\left(2L^{k+1}-L^k+2S^{k+1}-S^k+Z^{k+1}-M\right),\tag{3.3e}$$

$$\Lambda_2^{k+1}=\Lambda_2^k-\frac{1}{s}\left(K^{k+1}+L^k-2L^{k+1}\right).\tag{3.3f}$$

The simplicity of the above scheme is that all the subproblems have closed-form solutions. As we see, model (1.3) turns out to be (1.1) when we set $\sigma > 0$ and abandon the constraint $L \geq 0$. Under these circumstances, subproblems (3.3a) and (3.3b) are the solution of (1.1). We now show the reason that the four subproblems can be solved easily. The first subproblem (3.3a) is equivalent to solving the proximal mapping of the nuclear norm $\|L\|_*$ and can be expressed by

$$L^{k+1}:=\operatorname{MatShrink}\left(L^k+\frac{1}{r}\left(\Lambda_1^k-\Lambda_2^k\right),\frac{1}{r}\right),\tag{3.4}$$

where the matrix shrinkage operation $\operatorname{MatShrink}(M,\alpha)$ $(\alpha > 0)$ is defined as

$$\operatorname{MatShrink}(M,\alpha) := U\operatorname{Diag}\bigl(\max\{\sigma-\alpha,0\}\bigr)V^T,$$

and $U\operatorname{Diag}(\sigma)V^T$ is the SVD of the matrix M, see [17] and [18].

The S-subproblem (3.3b) can be solved by

$$S^{k+1} := \operatorname{Shrink}\left(S^k + \frac{1}{r}\Lambda_1^k, \frac{\rho}{r}\right), \tag{3.5}$$

where the ℓ_1 shrinkage operator [19] $\operatorname{Shrink}(M,\alpha)$ is defined as

$$\bigl[\operatorname{Shrink}(M,\alpha)\bigr]_{ij} := \begin{cases} M_{ij}+\alpha, & \text{if } M_{ij} < -\alpha, \\ M_{ij}-\alpha, & \text{if } M_{ij} > \alpha, \\ 0, & \text{if } |M_{ij}| \le \alpha. \end{cases}$$

The closed-form solution of the third subproblem (3.3c) can be written as

$$Z^{k+1} := W^k / \max\bigl\{1, \|W^k\|_F/\sigma\bigr\}, \tag{3.6}$$

which means projecting the matrix $W^k := M + s\Lambda_1^k - (2L^{k+1} - L^k + 2S^{k+1} - S^k)$ onto the Euclidean ball $\|Z\|_F \le \sigma$. The K-subproblem (3.3d) corresponds to projecting the matrix $(2L^{k+1} - L^k + s\Lambda_2^k - M)$ onto the nonnegative orthant and this can be done by

$$K^{k+1} := \max\bigl\{2L^{k+1} - L^k + s\Lambda_2^k - M, 0\bigr\}, \tag{3.7}$$

where the max function is componentwise [4].

4 Convergence analysis

In this section, we will show the global convergence result of the algorithm proposed for solving (2.1) or (3.2). First, we need to prove the following lemma.

Lemma 1 *Let $w^{k+1} = (x^{k+1}, y^{k+1}; \lambda^{k+1})$ be generated by the proposed algorithm from the given $w^k = (x^k, y^k; \lambda^k)$. Then we can have*

$$h(u) - h\bigl(u^{k+1}\bigr) + \bigl(w - w^{k+1}\bigr)^T\bigl(F\bigl(w^{k+1}\bigr) + G\bigl(w^{k+1} - w^k\bigr)\bigr) \ge 0, \quad \forall w \in \Omega, \tag{4.1}$$

where

$$G = \begin{pmatrix} I & 0 & 0 \\ 0 & rI & B^T \\ 0 & B & sI \end{pmatrix}.$$

Proof Deriving the first-order optimality condition of the first equality in Algorithm 1, we can obtain

$$g(y) - g\bigl(y^{k+1}\bigr) + \bigl(y - y^{k+1}\bigr)^T\bigl[-B^T\lambda^k + r\bigl(y^{k+1} - y^k\bigr)\bigr] \ge 0, \quad \forall y \in \mathcal{Y}. \tag{4.2}$$

It can also be expressed as

$$g(y)-g(y^{k+1})+(y-y^{k+1})^T\big[B^T(\lambda^{k+1}-\lambda^k)-B^T\lambda^{k+1}+r(y^{k+1}-y^k)\big]\geq 0,\quad \forall y\in\mathcal{Y}. \tag{4.3}$$

Homoplastically, the second iteration for solving x^{k+1} in Algorithm 1 shows us that

$$\begin{aligned} &f(x)-f(x^{k+1})+(x-x^{k+1})^T\bigg[-A\lambda^k+\frac{1}{s}A^T\big(Ax^k+B(2y^{k+1}-y^k)-b\big)\\ &\quad+\left(I+\frac{1}{s}A^TA\right)(x^{k+1}-x^k)\bigg]\geq 0,\quad \forall x\in\mathcal{X}. \end{aligned}\tag{4.4}$$

Substituting $\lambda^k=\lambda^{k+1}+\frac{1}{s}(Ax^{k+1}+B(2y^{k+1}-y^k)-b)$ into (4.4), we get

$$f(x)-f(x^{k+1})+(x-x^{k+1})^T\big(-A^T\lambda^{k+1}+(x^{k+1}-x^k)\big)\geq 0,\quad \forall x\in\mathcal{X}. \tag{4.5}$$

Note that the λ-iteration can be written as

$$Ax^{k+1}+By^{k+1}-b+B(y^{k+1}-y^k)+s(\lambda^{k+1}-\lambda^k)=0. \tag{4.6}$$

Merging (4.3), (4.5) and (4.6), we achieve

$$\begin{aligned} &h(u)-h(u^{k+1})+\begin{pmatrix} x-x^{k+1}\\ y-y^{k+1}\\ \lambda-\lambda^{k+1}\end{pmatrix}^T\\ &\quad\times\left\{\begin{pmatrix} -A^T\lambda^{k+1}\\ -B^T\lambda^{k+1}\\ Ax^{k+1}+By^{k+1}-b\end{pmatrix}+\begin{pmatrix} (x^{k+1}-x^k)\\ r(y^{k+1}-y^k)+B^T(\lambda^{k+1}-\lambda^k)\\ B(y^{k+1}-y^k)+s(\lambda^{k+1}-\lambda^k)\end{pmatrix}\right\}\geq 0,\quad \forall w\in\Omega. \end{aligned}\tag{4.7}$$

Utilizing the notation of w, $F(w)$ and the matrix G, the inequality above can be rewritten as

$$h(u)-h(u^{k+1})+(w-w^{k+1})^T\big(F(w^{k+1})+G(w^{k+1}-w^k)\big)\geq 0,\quad \forall w\in\Omega. \tag{4.8}$$

In other words, the lemma is proved. Inequality (4.8) implies that the proposed algorithm is in fact equivalent to the proximal point algorithm with a special proximal regularization parameter matrix G. □

Now we show and prove the contractive property of the proposed algorithm.

Lemma 2 *Assume that the parameters $r>0$ and $s>\frac{1}{r}\|B^TB\|$ are satisfied. Let w^{k+1} be the sequence generated by the new algorithm with an arbitrary initial iterate w^0. Then it holds*

$$\|w^{k+1}-w^*\|_G^2\leq\|w^k-w^*\|_G^2-\|w^{k+1}-w^k\|_G^2,\quad \forall w^*\in\Omega^*. \tag{4.9}$$

The norm $\|\cdot\|_G^2$ is defined as $\|w\|_G^2=\langle w,Gw\rangle$ and the corresponding inner product $\langle\cdot,\cdot\rangle_G$ is defined as $\langle u,v\rangle_G=\langle u,Gv\rangle$.

Proof Because $w^* \in \Omega^*$ is optimal to (2.1), it follows from the KKT conditions that the following hold:

$$0 \in \partial f(x^*) - A^T\lambda^*, \tag{4.10a}$$

$$0 \in \partial g(y^*) - B^T\lambda^*, \tag{4.10b}$$

$$0 = Ax^* + By^* - b. \tag{4.10c}$$

As we see, the optimality condition for the first subproblem in Algorithm 1 is

$$0 \in \partial g(y^{k+1}) + r\left(y^{k+1} - y^k - \frac{1}{r}B^T\lambda^k\right). \tag{4.11}$$

Combining (4.10b) and (4.11) under the fact that $\theta(\cdot)$ is monotone, we have

$$(y^{k+1} - y^*)^T\left(B^T\lambda^k - r(y^{k+1} - y^k) - B^T\lambda^*\right) \geq 0. \tag{4.12}$$

Similarly, the optimality condition for the subproblem with respect to x can be given by

$$\begin{aligned}0 \in \partial f(x^{k+1}) - A^T\left(\lambda^k - \frac{1}{s}\left(Ax^{k+1} + B(2y^{k+1} - y^k) - b\right)\right)\\ + \frac{1}{s}A^TA(x^{k+1} - x^k) + (x^{k+1} - x^k).\end{aligned} \tag{4.13}$$

Substituting the λ-subproblem into (4.12), we obtain

$$0 \in \partial f(x^{k+1}) - A^T\lambda^{k+1} + (x^{k+1} - x^k). \tag{4.14}$$

Combining (4.10a) and (4.13), we get

$$(x^{k+1} - x^*)^T\left(A^T\lambda^{k+1} - (x^{k+1} - x^k) - A^Tx^*\right) \geq 0. \tag{4.15}$$

Summing (4.12) and (4.15), we can achieve

$$\begin{aligned}&(x^{k+1} - x^*)^TA^T(\lambda^{k+1} - \lambda^*) + (x^{k+1} - x^*)^T(x^k - x^{k+1}) + (y^{k+1} - y^*)^TB^T(\lambda^k - \lambda^{k+1})\\ &\quad + (y^{k+1} - y^*)^TB^T(\lambda^{k+1} - \lambda^*) + r(y^{k+1} - y^*)^T(y^k - y^{k+1}) \geq 0.\end{aligned} \tag{4.16}$$

Combining the λ-subproblem with (4.10c), we get

$$(\lambda^{k+1} - \lambda^*)^T\left(B(y^k + y^* - 2y^{k+1}) - A(x^{k+1} - x^*) + s(\lambda^k - \lambda^{k+1})\right) \geq 0. \tag{4.17}$$

Note that (4.17) can be rewritten as

$$\begin{aligned}&(\lambda^{k+1} - \lambda^*)^TB(y^k - y^{k+1}) + s(\lambda^{k+1} - \lambda^*)^T(\lambda^k - \lambda^{k+1})\\ &\quad \geq (x^{k+1} - x^*)^TA^T(\lambda^{k+1} - \lambda^*) + (y^{k+1} - y^*)^TB^T(\lambda^{k+1} - \lambda^*).\end{aligned} \tag{4.18}$$

Using the definition of $\langle\cdot,\cdot\rangle_G$ and (4.18), we have

$$\begin{aligned}&\langle w^{k+1}-w^*, w^k-w^{k+1}\rangle_G\\&\quad\geq (x^{k+1}-x^*)^T A^T(\lambda^{k+1}-\lambda^*)+(x^{k+1}-x^*)^T(x^k-x^{k+1})\\&\qquad+(y^{k+1}-y^*)^T B^T(\lambda^k-\lambda^{k+1})+r(y^{k+1}-y^*)^T(y^k-y^{k+1})\\&\qquad+(y^{k+1}-y^*)^T B^T(\lambda^{k+1}-\lambda^*).\end{aligned}\tag{4.19}$$

Recall (4.16), we can easily get

$$\langle w^{k+1}-w^*, w^k-w^{k+1}\rangle_G\geq 0.\tag{4.20}$$

Therefore

$$\langle w^{k+1}-w^k, w^*-w^k\rangle_G\geq \|w^{k+1}-w^k\|_G^2.\tag{4.21}$$

Combining (4.21) with the identity

$$\|w^{k+1}-w^*\|_G^2=\|w^{k+1}-w^k\|_G^2-2\langle w^{k+1}-w^k, w^*-w^k\rangle_G+\|w^k-w^*\|_G^2,\tag{4.22}$$

we get

$$\|w^{k+1}-w^*\|_G^2\leq\|w^k-w^*\|_G^2-\|w^{k+1}-w^k\|_G^2.\tag{4.23}$$

This completes the proof. Note that the sequence $\{w^k\}$ is Fejér monotone with respect to the solution set. In addition, the proposed algorithm to solve (2.1) or (3.2) has the framework of contraction type methods. Therefore, by using the Fejér monotonicity and the contractility, the rest of the convergence proof becomes standard. Here, we do not repeat. We refer the readers [20] for more details. □

5 Numerical results

In this section, we study the performance of Algorithm 1 for solving (1.3). Our codes were written in MATLAB R2009a. In addition, all of the experiments were performed on a laptop with an Intel Core 2 Duo CPU at 2.2 GHz and 2 GB memory. In the experiments, we get the data randomly in the same way as in [4]. For given n, $r<n$, the $n\times n$ matrix L^* with rank-r was generated by $R_1R_2^T$, where R_1 and R_2 are both random matrices with all components distributed in $[0,1]$ uniformly. As we see, L^* is a nonnegative and low-rank matrix we want to recover. The support of the sparse matrix S^* was chosen uniformly at random, and the nonzero components of S^* were drawn uniformly in the interval $[-500,500]$. The components of matrix Z^* for noise were generated as i.i.d. Gaussian with standard deviation 10^{-4}. Then we set $M=L^*+S^*+Z^*$. According to the suggestion in [2], we chose $\rho=1/\sqrt{n}$. The starting point for the two algorithms was set as $L^0=K^0=-M$, $S^0=Z^0=0$, $\Lambda_1^0=\Lambda_2^0=0$. In our experiments, we use

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

Table 1 Comparison of the CPU times between APGM and Algorithm 1

n	Algorithm	$R_r = 0.01, C_r = 0.01$	$R_r = 0.02, C_r = 0.02$	$R_r = 0.03, C_r = 0.03$
		min/avg/max	min/avg/max	min/avg/max
100	APGM	0.4/**0.5**/0.9	0.9/**1.1**/1.7	0.9/**1.1**/1.4
	Algorithm 1	0.7/**0.9**/1.6	1.0/**1.4**/2.3	1.1/**1.1**/1.2
150	APGM	1.5/**1.8**/2.0	2.2/**2.4**/2.6	2.1/**2.2**/2.4
	Algorithm 1	1.8/**2.4**/2.9	2.0/**2.2**/2.4	2.2/**2.3**/2.5
200	APGM	3.3/**4.0**/6.6	4.1/**4.7**/6.5	3.4/**4.0**/5.0
	Algorithm 1	3.6/**4.2**/5.4	3.8/**4.0**/4.9	4.0/**4.5**/5.3
250	APGM	6.3/**8.9**/18.5	6.2/**6.9**/8.5	5.4/**6.9**/9.7
	Algorithm 1	6.3/**8.0**/12.5	6.5/**6.7**/7.3	7.3/**9.6**/16.4
300	APGM	10.8/**11.7**/12.5	9.1/**10.5**/15.4	8.0/**9.8**/12.6
	Algorithm 1	10.5/**11.7**/14.0	10.5/**11.8**/15.6	11.5/**14.1**/16.1
400	APGM	33.6/**36.1**/38.6	28.9/**30.1**/31.2	29.7/**30.2**/33.5
	Algorithm 1	33.6/**36.0**/38.0	37.8/**39.7**/41.4	30.9/**45.1**/47.2
500	APGM	69.6/**74.9**/83.0	63.1/**66.0**/67.3	62.1/**64.7**/66.5
	Algorithm 1	79.8/**83.7**/93.9	86.8/**89.5**/91.6	90.7/**94.5**/97.6

Table 2 Comparison of the iteration numbers between APGM and Algorithm 1

n	Algorithm	$R_r = 0.01, C_r = 0.01$	$R_r = 0.02, C_r = 0.02$	$R_r = 0.03, C_r = 0.03$
		min/avg/max	min/avg/max	min/avg/max
100	APGM	30/**43**/70	62/**68**/74	67/ **72**/77
	Algorithm 1	49/**60**/72	56/**62**/73	60/ **63**/66
150	APGM	43/**53**/59	60/**63**/64	53/ **55**/57
	Algorithm 1	47/**57**/71	44/**48**/51	46/ **49**/51
200	APGM	45/**50**/54	51/**53**/57	43/ **45**/47
	Algorithm 1	42/**47**/59	42/**43**/44	44/ **44**/45
250	APGM	47/**51**/53	43/**45**/46	36/ **37**/38
	Algorithm 1	41/**44**/50	41/**41**/42	43/ **44**/44
300	APGM	46/**47**/48	39/**40**/41	34/ **34**/34
	Algorithm 1	39/**40**/40	41/**41**/42	44/ **44**/44
400	APGM	41/**43**/43	33/**33**/33	32/ **33**/35
	Algorithm 1	39/**40**/40	42/**42**/42	45/ **45**/46
500	APGM	36/**37**/38	33/**33**/33	33/ **33**/33
	Algorithm 1	40/**40**/40	43/**43**/43	45/ **45**/46

as the recursion terminal condition. The tolerance parameter ϵ_r here is chosen as 10^{-4}. In order to ensure the rank of L^* be $n * R_r$ and the cardinality of S^* be $n^2 * C_r$, we denote $R_r := r/n$ and $C_r := \text{cardinality}(S^*)/(n^2)$, respectively. For different cases of m, R_r and C_r, we focus on the iteration numbers and CPU times in the experiments. Then we define

$$rms_L := \frac{\|L - L^*\|_F}{k}, \qquad rms_S := \frac{\|S - S^*\|_F}{k} \tag{5.2}$$

as the root mean square error of the matrix $L(rms_L)$ and root mean square error of the sparse matrix $S(rms_S)$, respectively, where k is the current number of iterations. In order to increase the persuasiveness, we randomly created ten examples, so the results were averaged over ten runs. The numerical results about the CPU times and iteration numbers are presented in Table 1 and Table 2. As we can see, our algorithm shows competitive performance with APGM in most cases. In some cases, Algorithm 1 is attended to be more efficient than APGM. For example, when $n \in [150, 250]$, $R_r = 0.02$ and $C_r = 0.02$, Algorithm 1 needs less CPU times and much fewer iteration numbers than APGM.

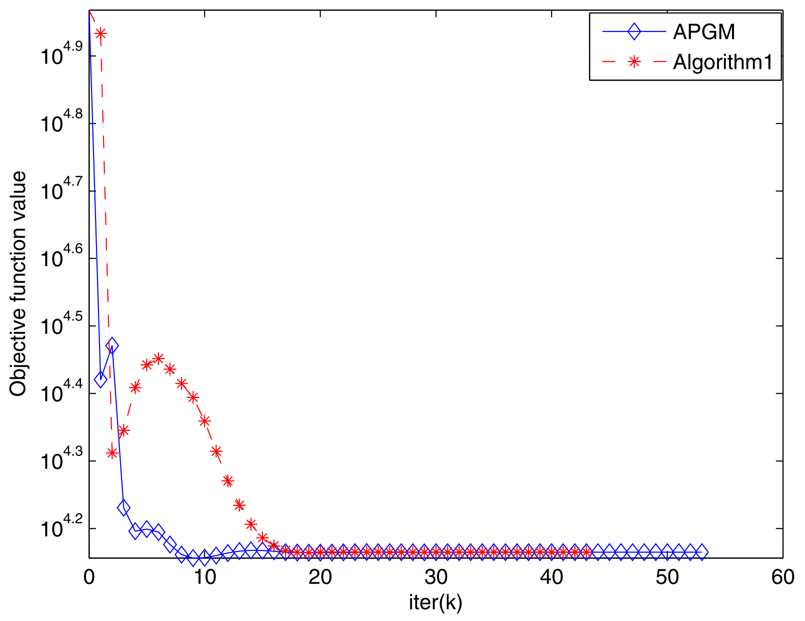

Figure 1 Comparison of objective function value with $n = 200$, $R_r = 0.02$, $C_r = 0.02$.

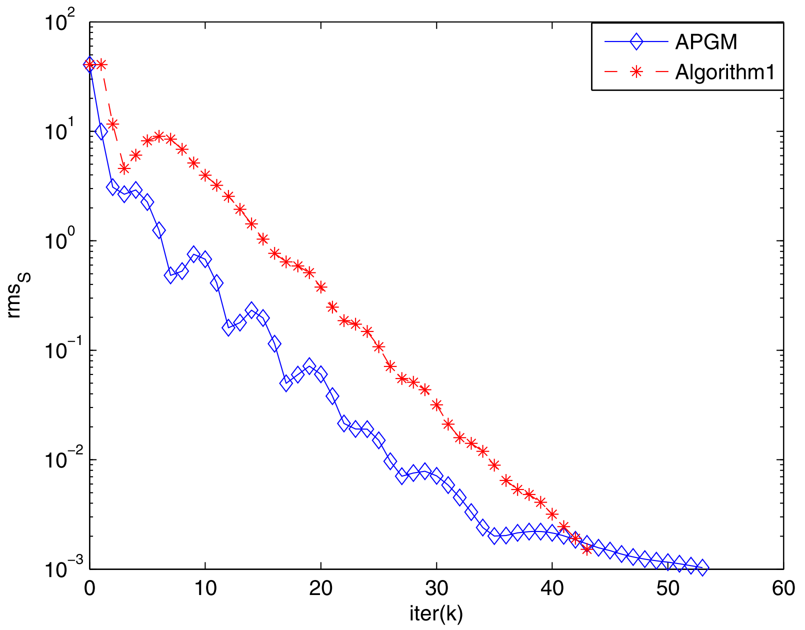

Figure 2 Comparison of rms_S with $n = 200$, $R_r = 0.02$, $C_r = 0.02$.

To better observe the convergence and performance of our algorithm, we plot the evolutions of the objective function value in Figure 1, rms_S in Figure 2 and rms_L in Figure 3, respectively. Plots in these figures indicate that the root mean squares of S and L decrease gently at first. However, when approaching the recursion terminal condition, the rms_S and rms_L in Algorithm 1 decrease more rapidly than APGM. In other words, Algorithm 1 meets the stopping criterion faster than APGM.

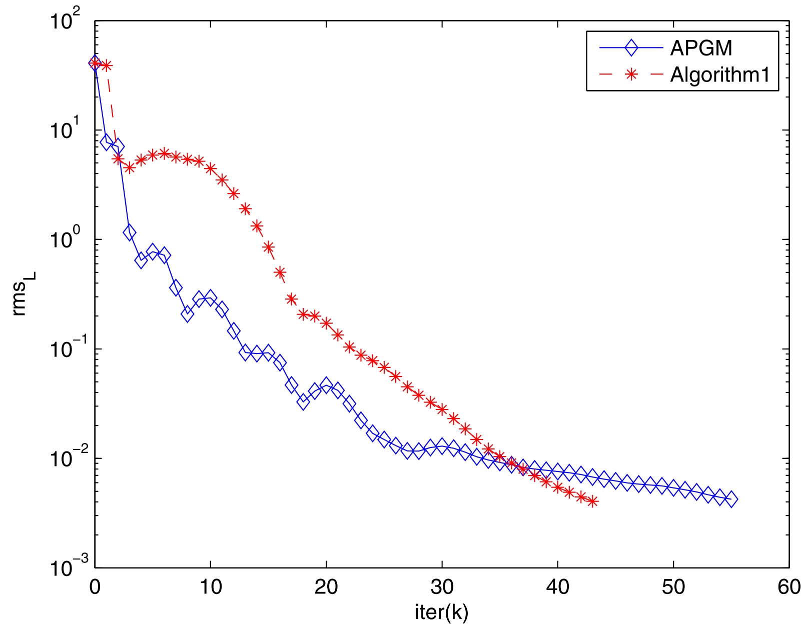

Figure 3 Comparison of rms_L with $n = 200$, $R_r = 0.02$, $C_r = 0.02$.

6 Conclusions

For solving the SPCP problem (1.3), we proposed a new algorithm based on the PPA in this paper. The global convergence of our algorithm is established. Then the computational results indicate that our algorithm achieves comparable performance with APGM. In certain circumstances, our algorithm can get better results than APGM.

Competing interests
The authors declare that they have no competing interests.

Authors' contributions
All the authors contributed equally. All authors read and approved the final manuscript.

Acknowledgements
The work is supported in part by the Natural Science Foundation of China Grant 71401176 and the Natural Science Foundation of Jiangsu Province Grant BK20141071.

Received: 3 December 2014 Accepted: 17 April 2015 Published online: 29 April 2015

References
1. Aybat, NS, Goldfarb, D, Ma, S: Efficient algorithms for robust and stable principal component pursuit problems. Comput. Optim. Appl. **58**(1), 1-29 (2014)
2. Candès, EJ, Li, X, Ma, Y, Wright, J: Robust principal component analysis? J. ACM **58**(3), 11 (2011)
3. Zhou, Z, Li, X, Wright, J, Candes, E, Ma, Y: Stable principal component pursuit. In: Information Theory Proceedings (ISIT), 2010 IEEE International Symposium on, pp. 1518-1522. IEEE Press, New York (2010)
4. Ma, S: Alternating proximal gradient method for convex minimization. Preprint (2012)
5. Aybat, NS, Iyengar, G: A unified approach for minimizing composite norms. Math. Program. **144**(1-2), 181-226 (2014)
6. Tao, M, Yuan, X: Recovering low-rank and sparse components of matrices from incomplete and noisy observations. SIAM J. Optim. **21**(1), 57-81 (2011)
7. Aybat, NS, Goldfarb, D, Iyengar, G: Fast first-order methods for stable principal component pursuit (2011). arXiv:1105.2126
8. He, B, Yuan, X: On the direct extension of ADMM for multi-block separable convex programming and beyond: from variational inequality perspective
9. He, B, Yuan, X, Zhang, W: A customized proximal point algorithm for convex minimization with linear constraints. Comput. Optim. Appl. **56**(3), 559-572 (2013)
10. Gu, G, He, B, Yuan, X: Customized proximal point algorithms for linearly constrained convex minimization and saddle-point problems: a unified approach. Comput. Optim. Appl. **59**(1-2), 135-161 (2014)

11. Boyd, S: EE364b Course Notes: Sub-Gradient Methods. Stanford University, Stanford, CA (2010)
12. Martinet, B: Brève communication. Régularisation d'inéquations variationnelles par approximations successives. ESAIM: Math. Model. Numer. Anal. **4**(R3), 154-158 (1970)
13. Rockafellar, RT: Augmented Lagrangians and applications of the proximal point algorithm in convex programming. Math. Oper. Res. **1**(2), 97-116 (1976)
14. Ma, F, Ni, M, Zhu, L, Yu, Z: An implementable first-order primal-dual algorithm for structured convex optimization. Abstr. Appl. Anal. **2014**, Article ID 396753 (2014)
15. 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), 1-122 (2011)
16. Chen, C, He, B, Ye, Y, Yuan, X: The direct extension of ADMM for multi-block convex minimization problems is not necessarily convergent. Math. Program., 1-23 (2014)
17. Ma, S, Goldfarb, D, Chen, L: Fixed point and Bregman iterative methods for matrix rank minimization. Math. Program. **128**(1-2), 321-353 (2011)
18. Cai, J-F, Candès, EJ, Shen, Z: A singular value thresholding algorithm for matrix completion. SIAM J. Optim. **20**(4), 1956-1982 (2010)
19. Parikh, N, Boyd, S: Proximal algorithms. Found. Trends Optim. **1**(3), 123-231 (2013)
20. Ma, S, Xue, L, Zou, H: Alternating direction methods for latent variable Gaussian graphical model selection. Neural Comput. **25**(8), 2172-2198 (2013)