A novel parameterized proximal point algorithm with applications in statistical learning

Jianchao Bai\textsuperscript{1}\textsuperscript{\dagger} Jicheng Li\textsuperscript{1} Jiaofen Li\textsuperscript{2}

\textsuperscript{1}School of Mathematics and Statistics, Xi’an Jiaotong University, Xi’an 710049, P.R. China
\textsuperscript{2}College of Mathematics and Computational Science, Guilin University of Electronic Technology, Guilin 541004, P.R. China

Abstract

In the literature, there are a few researches for the proximal point algorithm (PPA) with some parameters designed in the metric proximal matrix, especially for the multi-objective optimization problems. Introducing some parameters to the PPA can make it more attractive and flexible. By using the unified framework of the classical PPA and constructing a parameterized proximal matrix, in this paper, we develop a general parameterized PPA with a relaxation step for solving the multi-block separable convex programming problem. By making use of the variational inequality and some mathematical identities, the global convergence and worst-case $O(1/t)$ convergence rate of the proposed algorithm are established. Some preliminary numerical experiments on solving a sparse matrix minimization problem from statistical learning show that our new algorithm can be very efficient and robust compared with some state-of-the-art algorithms.

Keywords: Convex programming; Proximal point algorithm; Relaxation step; Variational inequality; Complexity; Statistical learning

AMS subject classifications(2010): 65C60; 65Y20; 90C25

1 Introduction

Throughout this paper, let $\mathbb{R}(\mathbb{R}^+), \mathbb{R}^m, \mathbb{R}^{m \times n}$ be the set of real (positive) numbers, the set of $m$ dimensional real column vectors and the set of $m \times n$ dimensional real matrices, respectively. The symbol $\|z\|_2$ denotes the Euclidean norm of $z \in \mathbb{R}^m$, which is defined by $\|z\|_2 = \sqrt{\langle z, z \rangle}$ where $\langle \cdot, \cdot \rangle$ means the inner product. For any symmetric positive definite matrix $G \in \mathbb{R}^{m \times m}$, we use $\|z\|_G = \sqrt{\langle z, Gz \rangle}$ to denote the weighted G-norm of $z$. The superscript $^T$ stands for the transpose, and the symbol $I$ represents the identity matrix with proper dimensions in the context.

We consider the following linearly constrained multi-block separable convex programming model

\[
\begin{align*}
\min & \quad \sum_{i=1}^{p} f_i(x_i) \\
\text{s.t.} & \quad \sum_{i=1}^{p} A_i x_i = b, \quad x_i \in \mathcal{X}_i,
\end{align*}
\]  

(1.1)
where \( p > 1 \) is a positive integer, all functions \( f_i(\cdot) : \mathbb{R}^{n_i} \to \mathbb{R} \) are closed convex (possibly nonsmooth); \( A_i \in \mathbb{R}^{m_i \times n_i}, b \in \mathbb{R}^m \) are given matrix and vector, respectively; all \( X_i \subset \mathbb{R}^{n_i} (i = 1, \cdots, p) \) are closed convex subsets. Throughout the discussions, we use the following basic assumption:

**Assumption 1.1** The solution set of the problem (1.1) is nonempty and all the matrices \( A_i (i = 1, \cdots, p) \) have full column rank.

Note that Assumption 1.1 is standard and formal, where the first part is basic and the second part is necessary. For instance, when solving a minimization problem subject to \( Ax = b \) involving large-size coefficient matrix and variable, we can split \( A = [A_1, \cdots, A_p] \) and \( x = (x^T_1, \cdots, x^T_p)^T \) to reduce the dimensions of \( A \) and \( x \), then each of \( A_i \) have full column rank. In fact, many other application problems do contain more than one variable, e.g. the total-variational image deblurring problems [10, 18], the sparse inverse covariance estimation problem in statistics [5], the low-rank and sparse problem in background modeling [11, 17] and so forth.

The proximal point algorithm (PPA), which was originally proposed to tackle the monotone operator inclusion problems [12, 13], is regarded as a kind of benchmark algorithms for solving the convex problems similar to (1.1). As verified by Rockafellar [16], the well-known augmented Lagrangian method for the problem (1.1) with \( p = 2 \) was actually an application of the PPA to its dual problem. Moreover, the classical alternating direction method of multipliers can be also treated as a special variant of the PPA to the dual problem [4]. In the last several years, the PPA had been extensively investigated by some researchers. For example, He et al.[7] showed a customized application of the classical PPA to the model (1.1) with \( p = 1 \), where some image processing problems were carried out to show the effectiveness of their method. Ma and Ni [14] also proposed a parameterized PPA for (1.1) with \( p = 1 \), where the basis pursuit problem and the matrix completion problem in numerical experiments were tested to indicate the performance of their algorithm. Cai et al.[2] designed a PPA with relaxation step for solving the model (1.1) with \( p = 2 \), where the global convergence and worst-case linear convergence rate of the algorithm therein were analyzed in detail. More recently, by introducing some parameters to the proximal matrix of the PPA, an extended parameterized PPA based on [14] was developed for solving the two block separable convex programming [1], and its efficiency was demonstrated by testing a sparse vector optimization problem arising in the statistical learning compared with two popular algorithms.

To the best of our knowledge, there are a few researches on the parameterized PPA for solving the multi-block separable convex programming (1.1) with \( p > 2 \). Based on such observation and mainly motivated by our recent work [1], the aims of this paper are to design a general parameterized PPA with a relaxation step (GR-PPA) for solving (1.1) and to test some practical examples with more than two variables to investigate the performance of the GR-PPA. In the remaining parts, Section 2 shows the details of constructing the GR-PPA and analyzing its global convergence and convergence rate in ergodic sense. In Section 3, some numerical examples are performed and comparative experiments are also carried out. Finally, we conclude and discuss the paper in Section 4.

## 2 Main results

In this section, we first construct a parameterized proximal matrix to design the GR-PPA for solving (1.1), where its convergence is analyzed in detail afterwards. The whole convergence analysis is based on the variational inequality and uses some special techniques to simplify the proof.

### 2.1 Formation of GR-PPA

For any \( \tau \in \mathbb{R}^+ \), the Lagrangian function of (1.1) can be constructed as

\[
L(x_1, \cdots, x_p, \lambda) = \sum_{i=1}^{p} f_i(x_i) - \tau \left\langle \lambda, \sum_{i=1}^{p} A_i x_i - b \right\rangle, \tag{2.1}
\]
where $\lambda \in \mathbb{R}^m$ is the Lagrange multiplier with respect to the equality constraint. Let $(x^*_1, \ldots, x^*_p, \lambda^*)$ be the saddle-point belonging to the solution set $\Omega^*$ of (1.1). Then, the following basic inequalities

$$L(x^*_1, \ldots, x^*_p, \lambda) \leq L(x^*_1, \ldots, x^*_p, \lambda^*) \leq L(x^*_1, \ldots, x^*_p, \lambda^*),$$

imply

$$\begin{align*}
x^*_1 &= \arg\min_{x_1 \in \mathcal{X}_1} \{ f_1(x_1) - \tau \langle \lambda^*, A_1 x_1 \rangle \}, \\
& \vdots \\
x^*_p &= \arg\min_{x_p \in \mathcal{X}_p} \{ f_p(x_p) - \tau \langle \lambda^*, A_p x_p \rangle \}, \\
\lambda^* &= \arg\max_{\lambda \in \mathbb{R}^m} -\tau \left( \lambda, \sum_{i=1}^{p} A_i x_i - b \right),
\end{align*}$$

whose optimality conditions are derived as follows

$$\begin{align*}
x^*_1 \in \mathcal{X}_1, & \quad f_1(x_1) - f_1(x^*_1) + \langle x_1 - x^*_1, -\tau A_1^T \lambda^* \rangle \geq 0, \quad \forall \, x_1 \in \mathcal{X}_1, \\
& \vdots \\
x^*_p \in \mathcal{X}_p, & \quad f_p(x_p) - f_p(x^*_p) + \langle x_p - x^*_p, -\tau A_p^T \lambda^* \rangle \geq 0, \quad \forall \, x_p \in \mathcal{X}_p, \\
\lambda^* \in \mathbb{R}^m, & \quad \langle \lambda - \lambda^*, \tau \left( \sum_{i=1}^{p} A_i x^*_i - b \right) \rangle \geq 0, \quad \forall \, \lambda \in \mathbb{R}^m.
\end{align*}$$

It is not hard to reformulate the above optimality conditions into a variational inequality:

$$w^* \in \Omega^*, \quad \text{VI}(\phi, \mathcal{J}, \Omega) : \phi(u) - \phi(u^*) + \langle w - w^*, \mathcal{J}(w^*) \rangle \geq 0, \quad \forall \, w \in \Omega,$$

where

$$\phi(u) = \sum_{i=1}^{p} f_i(x_i), \quad \Omega = \mathcal{X}_1 \times \cdots \times \mathcal{X}_p \times \mathbb{R}^m,$$

$$u = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{pmatrix}, \quad w = \begin{pmatrix} x_1 \\ \vdots \\ x_p \\ \lambda \end{pmatrix} \quad \text{and} \quad \mathcal{J}(w) = \tau \begin{pmatrix} -A_1^T \lambda \\ \vdots \\ -A_p^T \lambda \\ \sum_{i=1}^{p} A_i x_i - b \end{pmatrix}.$$ 

In the following, we also denote

$$u^k = \begin{pmatrix} x^k_1 \\ x^k_2 \\ \vdots \\ x^k_p \end{pmatrix}, \quad w^k = \begin{pmatrix} x^k_1 \\ \vdots \\ x^k_p \\ \lambda^k \end{pmatrix} \quad \text{and} \quad \mathcal{J}(w^k) = \tau \begin{pmatrix} -A_1^T \lambda^k \\ \vdots \\ -A_p^T \lambda^k \\ \sum_{i=1}^{p} A_i x^k_i - b \end{pmatrix}. \quad (2.3)$$

Using the property of the above skew-symmetric mapping $\mathcal{J}(w)$, we get

$$\langle w^{k+1} - w, \mathcal{J}(w^{k+1}) \rangle = \langle w^{k+1} - w, \mathcal{J}(w) \rangle, \quad \forall \, w, w^{k+1} \in \Omega. \quad (2.4)$$

It is well-known that the standard PPA with given iterate $w^k$ reads the following unified framework:

$$w^{k+1} \in \Omega, \quad \phi(u) - \phi(u^{k+1}) + \langle w - w^{k+1}, \mathcal{J}(w^{k+1}) + G \left( w^{k+1} - w^k \right) \rangle \geq 0, \quad \forall \, w \in \Omega, \quad (2.5)$$

where $G$ is symmetric positive definite matrix (called the proximal matrix). Next, a parameterized proximal matrix is constructed to develop the GR-PPA for solving the problem (1.1).
Concisely, we define the matrix $G$ in (2.5) as the following block form

$$
G = \begin{pmatrix}
\left(\sigma_1 + \frac{\varepsilon - 1}{s}\right)A_1^T A_1 & \left(\sigma_2 + \frac{\varepsilon - 1}{s}\right)A_2^T A_2 & \cdots & \left(\sigma_p + \frac{\varepsilon - 1}{s}\right)A_p^T A_p \\
-\varepsilon A_1 & -\tau A_2 & \cdots & -\tau A_p \\
\end{pmatrix},
$$

(2.6)

where $(\sigma_1, \cdots, \sigma_p, s)$ are parameters restricted into the domain

$$
\mathcal{K} = \left\{\sigma_1 > \frac{1 + (p - 1)\varepsilon}{s}, \sigma_i > \frac{1 + (p - 2)\tau^2 + \varepsilon}{s}, s > 0 \mid \varepsilon, \tau \in \mathbb{R}^+, i = 2, \cdots, p\right\}.
$$

(2.7)

For the sake of analysis convenience, here and next we denote

$$
\bar{\sigma}_i = \sigma_i + \frac{\tau^2 - 1}{s}, \forall i = 1, \cdots, p.
$$

(2.8)

**Lemma 2.1** Suppose that the second part of Assumption 1.1 holds. Then, for any $(\sigma_1, \cdots, \sigma_p, s) \in \mathcal{K}$, the matrix $G$ defined in (2.6) is symmetric positive definite.

**Proof.** Notice that the matrix $G$ is symmetric and can be decomposed as

$$
G = D^T G_0 D,
$$

where $D = \text{Diag}(A_1, \cdots, A_p, I)$ and

$$
G_0 = \begin{pmatrix}
\left(\sigma_1 + \frac{\varepsilon - 1}{s}\right)I & \left(\sigma_2 + \frac{\varepsilon - 1}{s}\right)I & \cdots & \left(\sigma_p + \frac{\varepsilon - 1}{s}\right)I \\
-\varepsilon I & -\tau I & \cdots & -\tau I \\
\end{pmatrix},
$$

(2.9)

Since all the matrices $A_i$ are assumed to have full column rank, therefore, the matrix $G$ is positive definite if and only if $G_0$ is positive definite. Using the following identity

$$
\begin{pmatrix}
I & \frac{\varepsilon}{s} I & \cdots & \frac{\varepsilon}{s} I \\
\vdots & \vdots & \ddots & \vdots \\
\frac{\varepsilon}{s} I & \vdots & \cdots & \frac{\varepsilon}{s} I \\
\end{pmatrix} G_0 
\begin{pmatrix}
I & \frac{\varepsilon}{s} I & \cdots & \frac{\varepsilon}{s} I \\
\vdots & \vdots & \ddots & \vdots \\
\frac{\varepsilon}{s} I & \vdots & \cdots & \frac{\varepsilon}{s} I \\
\end{pmatrix}^T = \begin{pmatrix}
(\sigma_1 - \frac{1}{s})I & -\frac{\varepsilon}{s} I & \cdots & -\frac{\varepsilon}{s} I \\
-\frac{\varepsilon}{s} I & (\sigma_2 - \frac{1}{s})I & \cdots & -\frac{\varepsilon}{s} I \\
\vdots & \vdots & \ddots & \vdots \\
-\frac{\varepsilon}{s} I & -\frac{\varepsilon}{s} I & \cdots & (\sigma_p - \frac{1}{s})I \\
\end{pmatrix} =: \tilde{G}_0,
$$

(2.10)

it is easy to demonstrate that $G_0$ is positive definite if and only if $\tilde{G}_0$ is positive definite, which is guaranteed by the region $\mathcal{K}$ defined in (2.7) because in such case both the upper-left and lower-right block matrices of $\tilde{G}_0$ are positive definite. Thus we complete the proof. □
Remark 2.1  It follows from the equation (3.7) in [8] that the proximal matrix of their proposed algorithm can be decomposed as $G = P^T G_0 P$, where

$$
G_0 = \begin{bmatrix}
\nu I & -I & \cdots & -I & 0 \\
-I & \nu I & \cdots & -I & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
-I & -I & \cdots & \nu I & 0 \\
0 & 0 & \cdots & 0 & I
\end{bmatrix},
$$
(2.11)

$P = \text{Diag}(\sqrt{\beta} A_1, \ldots, \sqrt{\beta} A_p, I_n/\sqrt{\beta})$, $\beta > 0$ is a penalty parameter with respect to the equality constraint of (1.1) and $\nu \geq m - 1$ denotes the proximal parameter in their paper. Compared (2.10) to (2.11), if we set

$$(s, \tau, \varepsilon) := (1, 1, 1) \quad \text{and} \quad \sigma_i := \nu + 1, \quad \forall \ i = 1, 2, \cdots, p,$$

then it is clear that $\tilde{G}_0 = G_0$, which implies that the proximal matrix of [8] in essence is a special case of (2.6). A similar way can be used to analyze the proximal matrix in equation (8.6) in [6]. Such relations show that our parameterized proximal matrix is more flexible than some in the literature.

In what follows, the iterative principles of our GR-PPA are analyzed one by one. Substituting the $\tilde{G}_0$ in (2.12) to (2.11), we have

$$0 = R_\lambda \tau \left( \frac{p}{i=1} A_i x_i^{k+1} - b \right) - \varepsilon A_1 (x_1^{k+1} - x_1^k) - \tau \sum_{i=2}^p A_i (x_i^{k+1} - x_i^k) + s (\lambda^{k+1} - \lambda^k).$$

Then, it can be derived from the above equality that

$$\lambda^{k+1} = \lambda^k - \frac{1}{s} \left( (\tau - \varepsilon) A_1 x_1^{k+1} + \varepsilon A_1 x_1^k + \tau \sum_{i=2}^p A_i x_i^k - b \right).$$

Combining (2.5)-(2.6) and (2.12), we have

$$x_1^{k+1} \in A_1, \quad f_1(x_1) - f_1(x_1^{k+1}) + \langle x_1 - x_1^{k+1}, R_{x_1} \rangle \geq 0, \quad \forall \ x_1 \in X_1,$$

where

$$R_{x_1} = -\tau A_1^T \lambda^k - \varepsilon A_1^T (\lambda^{k+1} - \lambda^k) + \left( \sigma_1 + \frac{\tau + \varepsilon}{s} \right) A_1^T A_1 (x_1^{k+1} - x_1^k)$$
$$= \varepsilon A_1^T \lambda^k - (\tau + \varepsilon) A_1^T \left\{ \lambda^k - \frac{1}{s} \left( (\tau - \varepsilon) A_1 x_1^{k+1} + \varepsilon A_1 x_1^k + \tau \sum_{i=2}^p A_i x_i^k - b \right) \right\}$$
$$= -\tau A_1^T \lambda^k + \frac{\tau + \varepsilon}{s} A_1^T A_1 x_1^{k+1} \left( \sum_{i=2}^p A_i x_i^k - b \right) + \left( \sigma_1 + \frac{\tau + \varepsilon}{s} \right) A_1^T A_1 x_1^{k+1}$$
$$= -\tau A_1^T \lambda^k + \left( \sigma_1 + \frac{\tau + \varepsilon}{s} \right) A_1^T A_1 x_1^{k+1} + \frac{(\tau + \varepsilon) \tau}{s} A_1^T \left( \sum_{i=2}^p A_i x_i^k - b \right)$$
$$= -\tau A_1^T \lambda^k + \frac{\tau + \varepsilon}{s} A_1^T A_1 x_1^{k+1} \left( \sum_{i=2}^p A_i x_i^k - b \right)$$

with $\tilde{\sigma_1}$ defined in (2.8) is positive followed by (2.7) and

$$\tilde{\lambda}^k = \lambda^k - \frac{\tau + \varepsilon}{s} \left( \sum_{i=1}^p A_i x_i^k - b \right).$$

(2.14)
By (2.13) together with $R_{x_1}$, obviously, $x_1^{k+1}$ is the solution of the following problem

$$x_1^{k+1} = \arg \min_{x_1 \in X_1} \left\{ f_1(x_1) - \langle A_1 x_1, \tau \bar{\lambda}^k \rangle + \frac{\sigma_1}{2} \| A_1 (x_1 - x_1^k) \|^2 \right\}$$

$$= \arg \min_{x_1 \in X_1} \left\{ f_1(x_1) + \frac{\sigma_1}{2} \| A_1 (x_1 - x_1^k) - \frac{\tau}{\sigma_1} \bar{\lambda}^k \|^2 \right\}. \quad (2.15)$$

Here we get from (2.12) and (2.14) that

$$\bar{\chi}_1^{k+1} = \chi^{k+1} - \frac{\tau + \epsilon}{s} \left( \sum_{i=1}^p A_i x_i^{k+1} - b \right)$$

$$= \chi^k - \frac{1}{s} \left[ (\tau - \epsilon) A_1 x_1^{k+1} + \epsilon A_1 x_1^k + \tau \sum_{i=2}^p A_i x_i^k - b \tau \right] - \frac{\tau + \epsilon}{s} \left( \sum_{i=1}^p A_i x_i^{k+1} - b \right)$$

$$= \bar{\chi}^k + \frac{\tau + \epsilon}{s} \left( \sum_{i=1}^p A_i x_i^k - b \right) - \frac{\tau + \epsilon}{s} \left( \sum_{i=1}^p A_i x_i^{k+1} - b \right)$$

$$= \bar{\chi}^k - \frac{\tau + \epsilon}{s} \sum_{i=1}^p A_i (x_i^{k+1} - x_i^k) - \frac{1}{s} \left[ (\tau - \epsilon) A_1 (x_1^{k+1} - x_1^k) + \tau \left( \sum_{i=1}^p A_i x_i^k - b \right) \right]. \quad (2.16)$$

Analogously, for $i = 2, 3, \ldots, p$, it follows from (2.5)-(2.6) and (2.12) that

$$x_i^{k+1} \in X_i, \quad f_i(x_i) - f_i(x_i^{k+1}) + (x_i - x_i^{k+1}, R_{x_i}) \geq 0, \quad \forall x_i \in X_i, \quad (2.17)$$

where

$$R_{x_i} = -\tau A_i^T \chi^{k+1} - \tau A_i^T (\chi^{k+1} - \chi^k) + \left( \sigma_i + \frac{\tau^2 - 1}{\tau} \right) A_i^T A_i (x_i^{k+1} - x_i^k)$$

$$= \tau A_i^T \chi^k - 2 \tau A_i^T \left\{ \chi^k - \frac{1}{s} \left[ (\tau - \epsilon) A_1 x_1^{k+1} + \epsilon A_1 x_1^k + \tau \sum_{i=2}^p A_i x_i^k - b \tau \right] \right\}$$

$$+ \left( \sigma_i + \frac{\tau^2 - 1}{\tau} \right) A_i^T A_i (x_i^{k+1} - x_i^k)$$

$$= -\tau A_i^T \chi^{k+1} + \sigma_i A_i^T A_i (x_i^{k+1} - x_i^k)$$

and

$$\bar{\chi}_i^{k+1} = \chi^k - \frac{2}{s} \left[ (\tau - \epsilon) A_1 x_1^{k+1} + \epsilon A_1 x_1^k + \tau \sum_{i=2}^p A_i x_i^k - b \tau \right]$$

$$= \bar{\chi}^k + \frac{\tau + \epsilon}{s} \left( \sum_{i=1}^p A_i x_i^k - b \right) - \frac{1}{s} \left[ (\tau - \epsilon) A_1 (x_1^{k+1} - x_1^k) + \tau \left( \sum_{i=1}^p A_i x_i^k - b \right) \right]$$

$$= \bar{\chi}^k + \frac{\tau + \epsilon}{s} \left( \sum_{i=1}^p A_i x_i^k - b \right) - \frac{2(\tau - \epsilon)}{s} A_1 (x_1^{k+1} - x_1^k)$$

$$= \bar{\chi}^k - \frac{\tau - \epsilon}{s} \left[ 2A_1 (x_1^{k+1} - x_1^k) + \sum_{i=1}^p A_i x_i^k - b \right]. \quad (2.18)$$

Therefore, $x_i^{k+1}$ is the solution of the following problem

$$x_i^{k+1} = \arg \min_{x_i \in X_i} \left\{ f_i(x_i) - \langle A_i x_i, \tau \bar{\chi}_i^{k+1} \rangle + \frac{\sigma_i}{2} \| A_i (x_i - x_i^k) \|^2 \right\}$$

$$= \arg \min_{x_i \in X_i} \left\{ f_i(x_i) + \frac{\sigma_i}{2} \| A_i (x_i - x_i^k) - \frac{\tau}{\sigma_i} \bar{\chi}_i^{k+1} \|^2 \right\}, \quad i = 2, \ldots, p. \quad (2.19)$$
Consequently, by using $\tilde{w}^k = (\tilde{x}_1^k, \ldots, \tilde{x}_p^k, \tilde{\lambda}^k)$ to denote the output of the $x_i$-subproblems and lagrange multipliers with given iterate $w^k = (x_1^k, \ldots, x_p^k, \lambda^k)$, and by utilizing $w^{k+1} = (x_1^{k+1}, \ldots, x_p^{k+1}, \lambda^{k+1})$ to stand for the new iterate together with a relaxation step, then the algorithmic framework of the GR-PPA is described as follows.

**Algorithm 2.1** (GR-PPA for solving Problem (1.1))

1. Choose parameters $(\sigma_1, \ldots, \sigma_p, s) \in \mathcal{K}, \gamma \in (0, 2]$ and initialize $(x_1^0, \ldots, x_p^0, \lambda^0) \in \Omega$;
2. $r^0 = \sum_{i=1}^p A_i x_i^0 - b$, $\lambda^0 = \lambda_0 - \frac{\gamma + \varepsilon}{s} r^0$ by (2.14);
3. For $k = 0, 1, \ldots$, if not converge, do
4. $\tilde{x}_1^k = \arg\min_{x_1 \in X_1} \left\{ f_1(x_1) + \frac{s}{2} \left\| A_1(x_1 - x_1^k) - \frac{\gamma}{s} \lambda^k \right\|^2 \right\}$;
5. $r^k = \sum_{i=1}^p A_i x_i^k - b$, $\Delta x_1^k = \tilde{x}_1^k - x_1^k$;
6. $\tilde{\lambda}^{k+\frac{1}{2}} = \lambda^k - \frac{\gamma + \varepsilon}{s} \left( 2A_1 \Delta x_1^k + r^k \right)$;
7. Update the $x_i$-subproblem for $i = 2, \ldots, p$ by (2.19):
8. $\tilde{x}_i^k = \arg\min_{x_i \in X_i} \left\{ f_i(x_i) + \frac{s}{2} \left\| A_i(x_i - x_i^k) - \frac{\gamma}{s} \tilde{\lambda}^{k+\frac{1}{2}} \right\|^2 \right\}$;
9. $\tilde{x}_p^k = \arg\min_{x_p \in \Omega} \left\{ f_p(x_p) + \frac{s}{2} \left\| A_p(x_p - x_p^k) - \frac{\gamma}{s} \tilde{\lambda}^{k+\frac{1}{2}} \right\|^2 \right\}$;
10. $\Delta x_i^k = \tilde{x}_i^k - x_i^k$, $\forall i = 2, \ldots, p$;
11. $\tilde{\lambda}^k = \tilde{\lambda}^k - \frac{\gamma + \varepsilon}{s} \sum_{i=1}^p A_i \Delta x_i^k - \frac{1}{s} \left[ (\gamma - \varepsilon) A_1 \Delta x_1^k + r^k \gamma \right]$ by (2.16);
12. Relaxation step: $\begin{pmatrix} x_1^{k+1} \\ \vdots \\ x_p^{k+1} \end{pmatrix} = \begin{pmatrix} x_1^k \\ \vdots \\ x_p^k \end{pmatrix} + \gamma \begin{pmatrix} \Delta x_1^k \\ \vdots \\ \Delta x_p^k \end{pmatrix}$ and $\tilde{\lambda}^{k+1} = \tilde{\lambda}^k + \gamma (\tilde{\lambda}^k - \tilde{\lambda})$.

**Remark 2.2** Notice that the above steps 4-9 are actually the PPA updates, and Algorithm 2.1 is an extension of our proposed parameterized PPA for two-block separable convex problem [1] when forcing $\gamma = 1$. However, the domain of the parameters restricted into (2.7) is not a direct extension of the past. Besides, Algorithm 2.1 adopts a serial idea between $x_1$-subproblem and other subproblems, while the parallel idea is used among the $x_i$-subproblem ($i = 2, \ldots, p$). From the relaxation step of Algorithm 2.1, we have the following certain relationship

$$w^{k+1} - w^k = \gamma(\tilde{w}^k - w^k).$$

(2.20)

### 2.2 Convergence analysis of GR-PPA

This subsection analyzes the global convergence and the ergodic convergence rate of Algorithm 2.1. First, we present an important lemma described in the following.

**Lemma 2.2** The sequence $\{w^{k+1}\}$ generated by Algorithm 2.1 satisfies

$$\|w^{k+1} - w^*\|^2_G \leq \|w^k - w^*\|^2_G - \frac{2 - \gamma}{\gamma} \|w^k - w^{k+1}\|^2_G, \quad \forall w^* \in \Omega^*.$$  \hspace{1cm} (2.21)
Proof By Algorithm 2.1 and (2.5), we have
\[ \tilde{w}^k \in \Omega, \quad \phi(u) - \phi(\tilde{w}^k) + \langle w - \tilde{w}^k, \mathcal{J}(\tilde{w}^k) + G(\tilde{w}^k - w^k) \rangle \geq 0, \quad \forall \ w \in \Omega, \] (2.22)
which, by using (2.2) and (2.4) with setting \( w = w^* \), leads to
\[ \langle \tilde{w}^k - w^*, G(w^k - \tilde{w}^k) \rangle \geq 0. \] (2.23)

Then, it follows from (2.20) and (2.23) that
\[
\| w^k - w^* \|^2_G - \| w^{k+1} - w^* \|^2_G = 2\gamma \langle w^k - w^*, G(w^k - \tilde{w}^k) \rangle - \gamma^2 \| \tilde{w}^k - w^k \|^2_G \\
= 2\gamma \langle w^k - \tilde{w}^k + \tilde{w}^k - w^*, G(w^k - \tilde{w}^k) \rangle - \gamma^2 \| w^k - \tilde{w}^k \|^2_G \\
\geq \gamma(2 - \gamma) \| w^k - \tilde{w}^k \|^2_G + 2\gamma \langle \tilde{w}^k - w^*, G(w^k - \tilde{w}^k) \rangle \\
= \frac{2 - \gamma}{\gamma} \| w^k - w^{k+1} \|^2_G,
\]
which immediately completes the proof. □

Lemma 2.2 shows that the sequence \( \{w^{k+1} - w^*\} \) is strictly contractive under the \( G \)-norm. Moreover, the following global convergence theorem holds.

**Theorem 2.1** Under the Assumption 1.1, for any \((\sigma_1, \cdots, \sigma_p, s) \in \mathcal{K}\) and the sequence \( \{w^{k+1}\} \) generated by Algorithm 2.1, there exists a \( w^\infty \in \Omega^* \) such that
\[
\lim_{k \to \infty} w^{k+1} = w^\infty.
\] (2.24)

**Proof** See the proof of Theorem 1 [1]. □

In order to establish the convergence rate of Algorithm 2.1, we first characterize the solution set of \( \text{VI}(\phi, \mathcal{J}, \Omega) \) in (2.2), which were given by e.g. [10] in the following:

**Theorem 2.2** The solution set of \( \text{VI}(\phi, \mathcal{J}, \Omega) \) is convex and can be characterized as
\[
\Omega^* = \bigcap_{w \in \Omega} \{ \hat{w} \in \Omega | \phi(u) - \phi(\hat{w}) + \langle w - \hat{w}, \mathcal{J}(w) \rangle \geq 0 \}.
\]

**Theorem 2.3** For any \((\sigma_1, \cdots, \sigma_p, s) \in \mathcal{K}\), let
\[
w_t = \frac{1}{1 + t} \sum_{k=0}^{t} \tilde{w}^k \quad \text{and} \quad u_t = \frac{1}{1 + t} \sum_{k=0}^{t} \tilde{u}^k,
\]
where \( \{\tilde{w}^k\} \) is the iterative sequence of Algorithm 2.1. Then, under the Assumption 1.1 we have
\[
\phi(u_t) - \phi(u) + \langle w_t - w, \mathcal{J}(w) \rangle \leq \frac{1}{2\gamma(1 + t)} \| w^0 - w \|^2_G, \quad \forall \ w \in \Omega.
\] (2.25)

**Proof** Combining (2.22) and (2.4), we can get
\[
\phi(u) - \phi(\tilde{w}^k) + \langle w - \tilde{w}^k, \mathcal{J}(w) \rangle \geq \langle \tilde{w}^k - w, G(\tilde{w}^k - w^k) \rangle.
\] (2.26)
Meanwhile, it follows from (2.20) and the following identity
\[ 2(a - b, G(c - d)) = \|a - d\|_G^2 - \|a - c\|_G^2 + \|c - b\|_G^2 - \|d - b\|_G^2 \]
with substitutions \((\hat{w}^k, w, w^{k+1}, u^k) = (a, b, c, d)\) that
\[ \langle \hat{w}^k - w, G(\hat{w}^k - w^k) \rangle = \frac{1}{4} \langle \hat{w}^k - w, G(w^{k+1} - w^k) \rangle = \frac{1}{2\gamma}(\|\hat{w}^k - w^k\|_G^2 - \|\hat{w}^k - w^k + w^{k+1} - w^k\|_G^2) + \|w^{k+1} - w\|_G^2 - \|w^k - w\|_G^2, \]
where the first two terms
\[ \|\hat{w}^k - w^k\|_G^2 - \|\hat{w}^k - w^k + w^{k+1} - w^k\|_G^2 = \|\hat{w}^k - w^k\|_G^2 - \|\hat{w}^k - w^k + w^{k+1} + w^k\|_G^2 = \gamma(2 - \gamma)\|\hat{w}^k - w^k\|_G^2 \geq 0. \]-(2.27)

Combining (2.26)-(2.28), we deduce
\[ \phi(u) - \phi(\hat{u}) + \langle w - \hat{w}^k, J(w) \rangle + \frac{1}{2\gamma}\|w^k - w\|_G^2 \geq \frac{1}{2\gamma}\|w^{k+1} - w\|_G^2. \]
summing the above inequality over \(k = 0, 1, \ldots, t\), we obtain
\[ (1 + t)\phi(u) - \sum_{k=0}^{t} \phi(\hat{u}^k) + \left(1 + t\right)\langle w - \sum_{k=0}^{t} \hat{w}^k, J(w) \rangle + \frac{1}{2\gamma}\|w^0 - w\|_G^2 \geq 0, \]
which by the definitions of \(w_t\) and \(u_t\) results in
\[ \frac{1}{1 + t}\sum_{k=0}^{t} \phi(\hat{u}^k) - \phi(u) + \langle w_t - w, J(w) \rangle \leq \frac{1}{2\gamma(1 + t)}\|w^0 - w\|_G^2. \]-(2.29)

Because of the convexity of the function \(\phi(u)\) (since all \(f_i\) are assumed to be convex), the following inequality holds
\[ \phi(u_t) \leq \frac{1}{1 + t}\sum_{k=0}^{t} \phi(\hat{u}^k), \]
which, substituting it into (2.29), completes the proof. □

**Remark 2.3** Theorem 2.3 illustrates a worst-case \(O(1/t)\) convergence rate of Algorithm 2.1 in an ergodic sense. By the region \(\gamma \in (0, 2)\) in Algorithm 2.1 and the inequality (2.25), one may choose a larger value \(\gamma\) approximating to 2 so that the right-hand value of (2.25) is much smaller.

### 3 Numerical experiments

This section aims to investigate the performance of the proposed GR-PPA for solving a statistical learning problem. All experiments are simulated in MATLAB 7.10(R2010a) on a PC with Intel Core i5 processor(3.3GHz) with 4 GB memory.

#### 3.1 Test problem

We focus on the following Latent Variable Gaussian Graphical Model Selection (LVGGMS) problem arising from statistical learning [3, 15]:
\[
\begin{align*}
\min & \quad F(X, S, L) := \langle X, C \rangle - \log \det(X) + \nu\|S\|_1 + \mu \text{tr}(L) \\
\text{s.t.} & \quad X - S + L = 0, \quad L \succeq 0,
\end{align*}
\]-(3.1)
where \( C \in \mathbb{R}^{n \times n} \) is a given covariance matrix obtained from the sample variables, \( \nu \) and \( \mu \) are given positive weighting factors, \( \text{tr}(L) \) denotes the trace of matrix \( L \), \( \|S\|_1 = \sum_{i,j=1}^{n} |S_{ij}| \) stands for the \( l_1 \)-norm of matrix \( S \) and \( S_{ij} \) means its \( ij \)-th entry.

Clearly, the LVGGMS problem (3.1) can be regarded as a special case of (1.1). And by applying Algorithm 2, it is easy to write the three corresponding subproblems as the following

\[
\begin{align*}
\tilde{X}^k &= \arg \min_{X \in \mathbb{R}^{n \times n}} \left\{ \langle X, C \rangle - \log \det(X) + \frac{\sigma_1}{2} \left\| X - (X^k + \frac{\tau}{\sigma_1} \lambda^k) \right\|^2_F \right\}, \\
\tilde{S}^k &= \arg \min_{S \in \mathbb{R}^{n \times n}} \left\{ \nu \|S\|_1 + \frac{\sigma_2}{2} \left\| S - (S^k - \frac{\tau}{\sigma_2} \lambda^k + \frac{1}{2}) \right\|^2_F \right\}, \\
\tilde{L}^k &= \arg \min_{L \succeq 0} \left\{ \mu \text{tr}(L) + \frac{\sigma_3}{2} \left\| L - (L^k + \frac{\tau}{\sigma_3} \lambda^k + \frac{1}{2}) \right\|^2_F \right\}.
\end{align*}
\]

Observe that the above subproblems in (3.2) have closed formula solutions. According to the first-order optimality condition of the \( X \)-subproblem in (3.2), we derive

\[
0 = C - \tilde{X}^k - \frac{\sigma_1}{\tau} \left( \tilde{X}^k - \frac{\tau}{\sigma_1} \lambda^k \right) \\
\Leftrightarrow \sigma_1 \tilde{X}^2 + \left( C - \sigma_1 \tilde{X}^k - \tau \lambda^k \right) \tilde{X} = 0.
\]

Then, by using the eigenvalue decomposition

\[
UDiag(\rho)U^T = C - \sigma_1 \tilde{X}^k - \tau \lambda^k,
\]

where \( Diag(\rho) \) is a diagonal matrix with diagonal entries \( \rho_i (i = 1, \cdots, n) \), we get its explicit solution

\[
\tilde{X}^k = U \text{Diag}(\gamma) U^T,
\]

where \( Diag(\gamma) \) is the diagonal matrix with diagonal entries

\[
\gamma_i = -\rho_i + \sqrt{\rho_i^2 + 4\sigma_1}, \quad i = 1, 2, \cdots, n.
\]

Applying the soft shrinkage operator \( \text{Shrink}(\cdot, \cdot) \), see e.g.[17], the solution of the \( S \)-subproblem is

\[
\tilde{S}^k = \text{Shrink} \left( S^k - \frac{\tau}{\sigma_2} \lambda^k + \frac{1}{2}, \frac{\nu}{\sigma_2} \right).
\]

Besides, it is obvious that the \( L \)-subproblem in (3.2) is equivalent to

\[
L^{k+1} = \arg \min_{L \succeq 0} \frac{\sigma_3}{2} \left\| L - \tilde{L}^k \right\|^2_F = V \text{Diag} \left( \max \{ \tilde{\rho}, 0 \} \right) V^T,
\]

where \( \max \{ \tilde{\rho}, 0 \} \) is taken component-wise and \( V \text{Diag}(\tilde{\rho}) V^T \) is the eigenvalue decomposition of

\[
\tilde{L} = L^k + \frac{\tau \lambda^k + \frac{1}{2} - \mu I}{\sigma_3}.
\]

### 3.2 Numerical results

In this subsection, some numerical examples of the LVGGMS problem (3.1) are performed, where the Algorithm 2.1 ("GR-PPA") is used to compare with two state-of-the-art algorithms: the Proximal Jacobian Decomposition of the ALM [8] ("PJALM") and the splitting algorithm of the ALM [9] ("HTY").
For all experiments, the maximal number of iterations is set as 1000 times, and the following stopping conditions are simultaneously used:

\[
\begin{align*}
\text{IER}(k) &:= \max \left\{ \frac{\|X^k - X^{k-1}\|_F}{\|X^k\|_F}, \frac{\|S^k - S^{k-1}\|_F}{\|S^k\|_F}, \frac{\|L^k - L^{k-1}\|_F}{\|L^k\|_F} \right\} \leq \epsilon_1, \\
\text{OER}(k) &:= \frac{|F(X^k, S^k, L^k) - F^*|}{|F^*|} \leq \epsilon_2, \\
\text{CER}(k) &:= \frac{\|X^k - S^k + L^k\|_F}{\max\{1, \|X^k\|_F, \|S^k\|_F, \|L^k\|_F\}} \leq \epsilon_3,
\end{align*}
\]

where \(X^k, S^k, L^k\) are the \(k\)-th iterative values, \(F^*\) is the approximate optimal objective function value by running the PJALM after 1000 iterations and \((\epsilon_1, \epsilon_2, \epsilon_3)\) are given tolerances.

The parameters \((\nu, \mu) = (0.005, 0.05)\) in (3.1) is taken from [15] and the data matrix \(C\) is randomly generated by the MATLAB codes of Boyd’s homepage\(^1\) with \(n = 100\). The parameters\(^2\) of GR-PPA are

\[
(s_1, s_2, s_3, \tau, \varepsilon) = \left(0.178, 0.178, 0.178, 10, \frac{\sqrt{5} - 1}{2}, \frac{\sqrt{5} - 1}{2} \right), \quad \gamma = 1.8,
\]

the parameter \(\beta = 0.05\) for both PJALM and HTY, the parameter \(\mu = 2.01\) for HTY as suggested in [9], and the proximal parameter of PJALM is fixed as 2. Experimental results of these three algorithms under different tolerances are reported in Table 1, where we use the starting feasible values \((X^0, S^0, L^0, X^0) = (2I, 4I, 2I, 1)\) and the notations “IT”, “CPU” denote the number of iterations and the CPU time in seconds, respectively. Moreover, with fixed tolerances \((\epsilon_1, \epsilon_2, \epsilon_3) = (10^{-12}, 10^{-14}, 10^{-12})\), comparative evaluations of the IER, OER and CER against the number of iterations with different starting points are shown in Figs. 1-3.

It is clear from Table 1 that under higher tolerances, GR-PPA could perform significantly better than PJALM and HTY in both number of iterations and the CPU time, although the HTY could perform better than GR-PPA under lower tolerances. Besides, the comparative convergence curves depicted in Fig.1-3 show that GR-PPA converges faster than other two comparison algorithms for different starting feasible points, which illustrates that the performance of GR-PPA does not depend on the initial iterative values. Results of Table 1 and convergence curves in Figs. 1-3 demonstrate the efficiency and robustness of the proposed algorithm.

4 Conclusion and discussion

In this paper, we develop a relaxed parameterized proximal point algorithm for solving a class of separable convex minimization problems. The global convergence and a worst-case linear convergence rate of the algorithm are established. Numerical experiments on testing a sparse matrix minimization problem in statistical learning also verify that our proposed algorithm performs slightly better than two popular methods when properly choosing the relaxation factor and the parameters in the proximal matrix.

Note that, in Sec.3, we only take a three-block problem of (1.1) for an example to investigate the performance of our algorithm. For the problem with more than three variables, the parameters in the GR-PPA need to adjust afresh via experiments. From the framework of the PPA in (2.5), one may have other choices to construct a novel parameterized proximal matrix to develop the corresponding PPA only if it is symmetric positive definite. Besides, notice that the proposed algorithm is applicable to the separable

\(^1\)http://web.stanford.edu/~boyd/papers/admm/covsel/covsel_example.html.
\(^2\)They are restricted into \(K\) in (2.7) and chosen after adjusting a lot of values. Actually, we observe that \((0.168, 0.168, 0.168, 30, \sqrt{2}, \sqrt{2})\) can be also set as the values of the parameters since the experimental results are still slightly better than other two algorithms.
Fig. 1: Evaluations of CER, IER and OER with initial values \((\mathbf{X}^0, S^0, L^0, \lambda^0) = (2I, 4I, 2I, I)\).

Fig. 2: Evaluations of CER, IER and OER with initial values \((\mathbf{X}^0, S^0, L^0, \lambda^0) = (2I, 3I, I, 0)\).

Fig. 3: Evaluations of CER, IER and OER with initial values \((\mathbf{X}^0, S^0, L^0, \lambda^0) = (I, 2I, I, 0)\).
convex problem where the coefficient matrices in the linear constraint have full column rank. Hence, three naturally questions appear: (1) whether there exists a PPA for the non-separable optimization? (2) how to develop a PPA for solving the separable convex problem with nonlinear constraint? (3) If there is no restriction for the rank of coefficient matrices, can we still develop a corresponding PPA? These questions need further investigations.

References


<table>
<thead>
<tr>
<th>( \epsilon_1 ) ( \epsilon_2 = 10^{-10} ) ( / \epsilon_3 = 10^{-6} )</th>
<th>GR-PPA</th>
<th>PJALM</th>
<th>HTY</th>
</tr>
</thead>
<tbody>
<tr>
<td>IT</td>
<td>CPU</td>
<td>IER</td>
<td>OER</td>
</tr>
<tr>
<td>( 10^{-4} )</td>
<td>106</td>
<td>1.55</td>
<td>2.0315e-7</td>
</tr>
<tr>
<td>( \epsilon_2 (\epsilon_1 = 10^{-6} ) ( / \epsilon_3 = 10^{-6} )</td>
<td>IT</td>
<td>CPU</td>
<td>IER</td>
</tr>
<tr>
<td>( 10^{-4} )</td>
<td>93</td>
<td>1.25</td>
<td>9.6286e-7</td>
</tr>
<tr>
<td>( 10^{-9} )</td>
<td>95</td>
<td>1.34</td>
<td>7.5073e-7</td>
</tr>
<tr>
<td>( \epsilon_3 (\epsilon_1 = 10^{-5} ) ( / \epsilon_2 = 10^{-8} )</td>
<td>IT</td>
<td>CPU</td>
<td>IER</td>
</tr>
<tr>
<td>( 10^{-7} )</td>
<td>109</td>
<td>1.86</td>
<td>1.3501e-7</td>
</tr>
<tr>
<td>( 10^{-11} )</td>
<td>188</td>
<td>2.65</td>
<td>1.2211e-11</td>
</tr>
</tbody>
</table>