A new customized proximal point algorithm for linearly constrained convex optimization *

Binqian Jiang†, Zheng Peng†‡, Wenxing Zhu †

Abstract: In this paper, we propose a new customized proximal point algorithm for linearly constrained convex optimization problem, and further use it to solve the separable convex optimization problem with linear constraints. Which is different to the existing customized proximal point algorithms, the proposed algorithm does not involve any relaxation step but still ensure the convergence. We obtain the particular iteration schemes and the unified variational inequality where the parameter matrix is symmetric and positive semi-definite, then the global convergence and a worst-case convergence rate of the proposed method are proven under some mild assumptions. Finally some numerical experiments show that, the proposed method is valid and high efficient comparing with some existing state-of-the-art methods.

Keywords: Convex optimization; separable convex optimization; customized proximal point algorithm; global convergence; $O(1/k)$ convergence rate

1 Introduction

Proximal point algorithm (PPA for short) could be dated back to Moreau [1], which was extended by Rockafellar [2], Bertsekas and Tseng [3], and Kaplan and Tichatschke [4], etc. It was introduced to optimization community by Martinet [5]. The PPA possesses a robust convergence theory for very general problems in finite and infinite-dimensions (Kassay [6] and Gler [7]), and is also the basis for the splitting methods (Spingarn [8] and Han [9]).

Consider a convex minimization problem of the form

$$\min\{\theta(x)|Ax-b=0, x\in X\}, \tag{1.1}$$

where $\theta:\mathbb{R}^n\to\mathbb{R}$ is convex (not necessarily smooth) function, $A\in\mathbb{R}^{m\times n}$, $b\in\mathbb{R}^m$, and $X\subset\mathbb{R}^n$ is a bounded closed convex nonempty-set. The solution set of (1.1), denoted by X^*, is assumed to be nonempty.

The iterative scheme of the PPA is:

$$\begin{cases} \tilde{\lambda}^k = \lambda^k - \dfrac{1}{t}(Ax^k-b), \\ \tilde{x}^k = \arg\min\Big\{\theta(x)+\dfrac{r}{2}\Big\|(x-x^k)-\dfrac{1}{r}A^T\lambda^k\Big\|^2\Big|x\in X\Big\}. \end{cases} \tag{1.2}$$

*This work was supported by the Natural Science Foundation of China (Grant No. 11571074, 61672005), the Natural Science Foundation of FuJian Province (Grant No. 2015J01010).

†School of Mathematics and Computer Science, Fuzhou University, Fuzhou, 350108, China.

‡Corresponding author. E-mail: pzheng@fzu.edu.cn

where $\lambda \in \mathbb{R}^m$ is a Lagrange multiplier vector. Let $W := X \times R^m$ and $w := (x, \lambda) \in W$. By the optimality conditions of (1.2), we have

$$\theta(x) - \theta(x^{k+1}) + (w - w^{k+1})^T \Big\{ F(w^{k+1}) + P(w^{k+1} - w^k) \Big\} \geq 0, \quad \forall\, w \in W, \tag{1.3}$$

where

$$F(w) = \begin{pmatrix} -A^T\lambda \\ Ax - b \end{pmatrix}, \quad P = \begin{pmatrix} rI_n, & 0 \\ -A, & tI_m \end{pmatrix}. \tag{1.4}$$

He, Yuan and Zhang [10] showed that the PPA could not be necessarily convergent. They proposed a customized proximal point algorithm (CPPA) by choosing a parameter matrix Q instead of the matrix P in (1.4) , where

$$Q = \begin{pmatrix} rI_n & -A^T \\ -A & tI_m \end{pmatrix}.$$

To guarantee convergence, the matrix Q is required to be positive semi-definite. As a consequence, it requires $rt > \rho(A^TA)$, where $\rho(H)$ denotes the spectral radius of matrix H. The corresponding iterative scheme of CPPA first produces a prediction $\tilde{w}^k = (\tilde{x}^k, \tilde{\lambda}^k)$ via

$$\begin{cases} \tilde{\lambda}^k = \lambda^k - \dfrac{1}{t}(Ax^k - b), \\ \tilde{x}^k = \arg\min \Big\{ \theta(x) + \dfrac{r}{2} \Big\| (x - x^k) - \dfrac{1}{r} A^T(2\tilde{\lambda}^k - \lambda^k) \Big\|^2 \Big| x \in X \Big\}. \end{cases} \tag{1.5}$$

Then, the CPPA usually generates the new iterate $w^{k+1} = (x^{k+1}, \lambda^{k+1})$ by a simple relaxation step

$$w^{k+1} = w^k - \gamma(w^k - \tilde{w}^k), \tag{1.6}$$

where $\gamma \in (0, 2)$ is a relaxation factor. Various numerical experiments show that $\gamma > 1$ provides the fast convergence. For distinction in this paper, the CPPA (1.5)-(1.6) with $\gamma = 1$ is named as the original-CPPA or CPPA for short, and the CPPA with $\gamma > 1$ is named as the relax-CPPA or RCPPA for short.

On the other hand, the augmented Lagrangian method (ALM) is also a efficient method for convex optimization, see Hestenes [11] and Powell [12]. Rockafellar [13] showed that the ALM is exactly an application of the PPA to the dual problem of (1.1), owning the same iteration scheme (1.3) where P is replaced by a thin matrix

$$Q = \begin{pmatrix} 0, & \dfrac{1}{r} I_m \end{pmatrix}^T.$$

However, by He, Yuan and Zhang [10], the CPPA is more efficient than the ALM when the objective function $\theta(x)$ is simple in the sense that its resolvent operator of $\theta(x)$, defined by $(I + \frac{1}{r}\partial\theta)^{-1}$, has a close-form representation. Here, $\partial(\cdot)$ denotes the sub-differential of a convex function.

The augmented Lagrangian-based methods, especially some splitting forms including alternating direction method of multipliers (ADMM for short) and parallel splitting algorithm [14], are verified to be very efficient for separable convex optimization problem of the form

$$\min \Big\{ \theta_1(x) + \theta_2(y) \Big| Ax + By = b, x \in X, y \in Y \Big\}, \tag{1.7}$$

where $X \subset \mathbb{R}^{n_1}$ and $Y \subset \mathbb{R}^{n_2}$ are nonempty bounded closed convex sets, $\theta_1 : X \to \mathbb{R}$ and $\theta_2 : Y \to \mathbb{R}$ are convex (not necessarily smooth), $A \in \mathbb{R}^{m\times n_1}$, $B \in \mathbb{R}^{m\times n_2}$ and $b \in \mathbb{R}^m$. The iterative scheme of the ALM to the problem (1.7) is:

$$\begin{cases} \lambda^{k+1} = \lambda^k - \dfrac{1}{t}(Ax^k - b), \\ (x^{k+1}, y^{k+1}) = \arg \min\limits_{x\in X, y\in Y} \left\{ \theta_1(x) + \theta_2(y) + \dfrac{\beta}{2}\left\| Ax + By - b - \dfrac{\beta}{2}\lambda^{k+1} \right\|^2 \right\}. \end{cases} \tag{1.8}$$

In the second subproblem of the scheme (1.8), the coupled variable (x, y) makes it intractable. As a splitting version of the ALM, the ADMM proposed originally by Glowinski [15] and Gabay [16] overcomes the drawback by discoupling the variable (x, y). Gabay [17] expressed the ADMM as essentially an application of the Douglas-Rachford splitting method [18]. Eckstein and Bertsekas [19] proposed a generalized ADMM (GADMM) as follows:

$$\begin{cases} \tilde{x}^k = \operatorname{Arg\,min}\left\{ \theta_1(x) + \dfrac{\beta}{2}\left\| Ax + By^k - b - \dfrac{\beta}{2}\lambda^k \right\|^2 \middle| x \in X \right\}, \\ \tilde{\lambda}^k = \lambda^k - \beta(A\tilde{x}^k + By^k - b), \\ \tilde{y}^k = \operatorname{Arg\,min}\left\{ \theta_2(y) + \dfrac{\beta}{2}\left\| A\tilde{x}^k + By - b - \dfrac{\beta}{2}\tilde{\lambda}^k \right\|^2 \middle| y \in Y \right\}. \end{cases} \tag{1.9}$$

The iterative scheme (1.9) can be recovered by a variational inequality reformulation with the compact form:

$$\theta(u) - \theta(\tilde{u}^k) + (w - \tilde{w}^k)^T \left\{ F(\tilde{w}^k) + Q(\tilde{w}^k - w^k) \right\} \geq 0, \quad \forall\, w \in W, \tag{1.10}$$

where $W = X \times Y \times R^m$ and $\theta(u) = \theta_1(x) + \theta_2(y)$, and

$$u = \begin{pmatrix} x \\ y \end{pmatrix}, \quad w = \begin{pmatrix} x \\ y \\ \lambda \end{pmatrix}, \quad F(w) = \begin{pmatrix} -A^T\lambda \\ -B^T\lambda \\ Ax + By - b \end{pmatrix}, \quad Q = \begin{pmatrix} 0 & 0 \\ \beta B^T B & -B^T \\ -B & \frac{1}{\beta} I_m \end{pmatrix}. \tag{1.11}$$

The new iterate $w^{k+1} = (x^{k+1}, y^{k+1}, \lambda^{k+1})$ is generated by a relaxation step:

$$w^{k+1} = w^k - \beta(w^k - \tilde{w}^k). \tag{1.12}$$

Note that matrix Q is not square, which means that the intermediate variable x may not be involved in the iteration. The matrix

$$Q_1 = \begin{pmatrix} \beta B^T B & -B^T \\ -B & \frac{1}{\beta} I_m \end{pmatrix},$$

which removes the zero row from Q in (1.11), is also required to be symmetric and positive semi-definite to ensure the convergence of the GADMM. This method was verified to be faster than ADMM because the approximate computation was permitted. He, et al [20] presented a general relaxation step as follows:

$$w^{k+1} = w^k - \gamma M(w^k - \tilde{w}^k), \tag{1.13}$$

where γ is the step-length based on the descent direction $M(w^k - \tilde{w}^k)$. Let $H = Q_1 M^{-1}$ and $G = Q_1^T + Q_1 - \gamma M^T H M$. One can relax the conventional proposition on Q_1 but ensure symmetric and positive semi-definite property of G. He and Yuan [21, 22] had established the worst-case $O(1/t)$ convergence rate and non-ergodic convergence rate of the Douglas-Rachford alternating direction method of multipliers.

However, the relaxation step might be unacceptable or even not be permitted in some practical applications, although it could accelerate the algorithm and ensure the convergence. Therefore, the proposed algorithm in this paper is designed to accept with no relaxation, and it could still have the high numerical performance and ensure the convergence.

The main work is to extend the CPPA without relaxation step in the following two aspects: The CPPA for the problem (1.1) is generalized by changing the x-update formulation of (1.5) to

$$x^{k+1} = \arg\min \Big\{\theta(x) + \frac{r}{2}\Big\|(x - x^k) - \frac{1}{r}A^T[(1+\alpha)2\lambda^{k+1} - \alpha\lambda^k]\Big\|^2 \Big| x \in X\Big\}. \tag{1.14}$$

Then, the above method is further used to solve the separable convex optimization problem of the form (1.7), which is called extended CPPA (ECPPA for short).

The rest of the paper is organized as follows. Section 2 proposes two methods including the generalized CPPA (GCPPA) and its extended version (ECPPA). In Section 3, under some mild assumptions, the convergence of the proposed methods is proved, and a worst-case $O(1/k)$ convergence rate is established. In Section 4, some preliminary numerical results, compared with the CPPA and RCPPA, the ADMM and GADMM, are presented to show the high efficiency of the GCPPA and ECPPA. Section 5 concludes this paper with some final remarks.

2 The generalized customized proximal point algorithms

For the linear-constrained convex optimization problem (1.1), i.e.,

$$\min\{\theta(x)|Ax - b = 0, x \in X\}, \tag{2.1}$$

the generalized customized proximal point algorithm is proposed as follows.

Algorithm 2.1: Generalized customized proximal point algorithm, GCPPA

For a given w^k, the GCPPA produces the new iterate $w^{k+1} \in W$ via the following scheme:

$$\begin{cases} \lambda^{k+1} = \lambda^k - \dfrac{\alpha}{t}(Ax^k - b), \\ x^{k+1} = \arg\min \Big\{\theta(x) + \dfrac{r}{2}\Big\|(x - x^k) - \dfrac{1}{r}A^T[(1+\alpha)\lambda^{k+1} - \alpha\lambda^k]\Big\|^2 \Big| x \in X\Big\}, \end{cases} \tag{2.2}$$

where $\alpha \in (0, 1]$ is a constant.

By the optimality condition we have that, for the new iterate $w^{k+1} \in W$ generated by the GCPPA, and $\forall w \in W$, the following variational inequalities (VIs for short) hold

$$\begin{cases} \theta(x) - \theta(x^{k+1}) + (x - x^{k+1})^T\Big(-A^T[(1+\alpha)\lambda^{k+1} - \alpha\lambda^k] + r(x^{k+1} - x^k)\Big) \geq 0, \\ (\lambda - \lambda^{k+1})^T\Big[\alpha(Ax^k - b) + t(\lambda^{k+1} - \lambda^k)\Big] \geq 0. \end{cases} \tag{2.3}$$

The VIs (2.3) can be rewritten to a compact form:

$$\theta(u) - \theta(u^{k+1}) + (w - w^{k+1})^T(F_\alpha(w^{k+1}) + Q(w^{k+1} - w^k)) \geq 0, \quad \forall\, w \in W, \tag{2.4}$$

where

$$u = x, \quad F_\alpha(w) = \begin{pmatrix} -A^T\lambda \\ \alpha(Ax-b) \end{pmatrix}, \quad Q = \begin{pmatrix} rI_n & -\alpha A^T \\ -\alpha A & tI_m \end{pmatrix}. \tag{2.5}$$

It is easy to show that, if $\alpha = 1$, the GCPPA reduces to the CPPA without a relaxation step, i.e., the case of $\gamma = 1$ in (1.6).

The GCPPA can be further extended to the separable convex optimization (1.7), which is:

$$\min\Big\{\theta_1(x) + \theta_2(y) \;\Big|\; Ax + By = b,\ x \in X,\ y \in Y\Big\}. \tag{2.6}$$

Let $W = X \times Y \times R^m$, and $w = (x, y, \lambda) \in W$. Then, for the separable convex optimization problem (2.6), an extended version of the GCPPA is proposed as follows.

Algorithm 2.2: An extension of the GCPPA to separable convex optimization, ECPPA

For a given w^k, the ECPPA produces a new iterate $w^{k+1} \in W$ via the following scheme:

$$\begin{cases} \lambda^{k+1} = \lambda^k - \dfrac{\alpha}{t}(Ax^k + By^k - b), \\ x^{k+1} = \arg\min\limits_{x\in X}\Big\{\theta_1(x) + \dfrac{r}{2}\Big\|(x - x^k) - \dfrac{1}{r}A^T\big[(\alpha+1)\lambda^{k+1} - \alpha\lambda^k\big]\Big\|^2\Big\}, \\ y^{k+1} = \arg\min\limits_{y\in Y}\Big\{\theta_2(y) + \dfrac{s}{2}\Big\|(y - y^k) - \dfrac{1}{s}B^T\big[(\alpha+1)\lambda^{k+1} - \alpha\lambda^k\big]\Big\|^2\Big\}, \end{cases} \tag{2.7}$$

where $\alpha \in (0, 1]$ is a constant.

Let $\theta(u) = \theta_1(x) + \theta_2(y)$, and

$$u = \begin{pmatrix} x \\ y \end{pmatrix}, \quad F_\alpha(w) = \begin{pmatrix} -A^T\lambda \\ -B^T\lambda \\ \alpha(Ax + By - b) \end{pmatrix}. \tag{2.8}$$

Then w^{k+1} generated by ECPPA satisfies that

$$\theta(u) - \theta(u^{k+1}) + (w - w^{k+1})^T\Big\{F_\alpha(w^{k+1}) + Q(w^{k+1} - w^k)\Big\} \ge 0, \quad \forall\, w \in W, \tag{2.9}$$

where

$$Q = \begin{pmatrix} rI_{n_1} & 0 & -\alpha A^T \\ 0 & sI_{n_2} & -\alpha B^T \\ -\alpha A & -\alpha B & tI_m \end{pmatrix}. \tag{2.10}$$

If $\alpha = 1$, then ECPPA reduces to an immediate application (to the separable convex optimization problem) of the CPPA (1.5)-(1.6) with $\gamma = 1$. This application is named as the separable CPPA (SCPPA in short) for conveniences of quotation in this paper.

3 Convergence analysis

By the previous discussion, for a given w^k, the new iterate $w^{k+1} \in W$ generated by the GCPPA and ECPPA satisfies the same condition, that is:

$$\theta(u) - \theta(u^{k+1}) + (w - w^{k+1})^T\Big\{F_\alpha(w^{k+1}) + Q(w^{k+1} - w^k)\Big\} \ge 0, \quad \forall\, w \in W. \tag{3.1}$$

Hence, the GCPPA and ECPPA may have the same convergence properties. We will prove in this section that, if the sequence $\{w^k\}$ satisfies the condition (3.1) in which F_α is monotone and Q is symmetric and positive semi-definite, it is global convergent and has a worst-case $O(1/k)$ convergence rate.

By the optimality condition, if $x^* \in X$ is a solution of the problem (2.1), there exists $\lambda^* \in R^m$ such that

$$\left\{ \begin{array}{r} \theta(x) - \theta(x^*) - (x - x^*)^T (A^T \lambda^*) \geq 0, \quad \forall\, x \in X, \\ (\lambda - \lambda^*)^T (Ax^* - b) = 0, \quad \forall\, \lambda \in R^m. \end{array} \right. \tag{3.2}$$

If $(x^*, y^*) \in X \times Y$ is a solution of the problem (2.6), there exists $\lambda^* \in R^m$ such that

$$\left\{ \begin{array}{r} \theta_1(x) - \theta_1(x^*) - (x - x^*)^T (A^T \lambda^*) \geq 0, \quad \forall\, x \in X, \\ \theta_2(y) - \theta_2(y^*) - (y - y^*)^T (B^T \lambda^*) \geq 0, \quad \forall\, y \in Y, \\ (\lambda - \lambda^*)^T (Ax^* + By^* - b) = 0, \quad \forall\, \lambda \in R^m. \end{array} \right. \tag{3.3}$$

If $\alpha > 0$ is bounded away from 0, then the last equation of (3.2) is identical to

$$(\lambda - \lambda^*)^T \, \big[\alpha(Ax^* - b)\big] = 0, \quad \forall\, \lambda \in R^m, \tag{3.4}$$

and respectively, the last equation of (3.3) is identical to

$$(\lambda - \lambda^*)^T \, \big[\alpha(Ax^* + By^* - b)\big] = 0, \quad \forall\, \lambda \in R^m. \tag{3.5}$$

Hence, the solution sets of the problems (2.1) and (2.6) can be described in a uniform style, i.e.,

Definition 3.1. The solution set, denoted by $W^* \subset W$, consists of all $w^* \in W$ satisfying

$$\theta(u) - \theta(u^*) + (w - w^*)^T F_\alpha(w^*) \geq 0, \quad \forall\, w \in W. \tag{3.6}$$

For a fixed $\alpha > 0$, $F_\alpha(w)$ in (2.4) or (2.9) is a linear mapping thus is obviously monotone on $w \in W$. The positive semi-definite property of the matrix Q in (2.5) and (2.10) can be guaranteed by the following theorem.

Theorem 3.1. *The following assertions are true:*

1). The matrix Q in (2.5) is positive semi-definite on $W = X \times R^m$ if $r, t > 0$ and

$$rt \geq \alpha^2 \rho(AA^T). \tag{3.7}$$

2). The matrix Q in (2.10) is positive semi-definite on $W = X \times Y \times R^m$ if $r, s, t > 0$ and

$$rt \geq \beta_1 \alpha^2 \rho(AA^T), \quad st \geq \beta_2 \alpha^2 \rho(BB^T), \tag{3.8}$$

where $\beta_1, \beta_2 > 0$ and $\dfrac{1}{\beta_1} + \dfrac{1}{\beta_2} \leq 1$.

Proof. 1). For $\forall\, w = (x, \lambda) \in W := X \times R^m$ and

$$Q = \begin{pmatrix} rI_n & -\alpha A^T \\ -\alpha A & tI_m \end{pmatrix},$$

we have

$$\begin{aligned}w^TQw &= r\|x\|^2 - 2\alpha x^T(A^T\lambda) + t\|\lambda\|^2\\ &= \frac{1}{r}\left(\|rx\|^2 - 2\alpha(rx)^T(A^T\lambda) + \|\alpha A^T\lambda\|^2\right) + t\|\lambda\|^2 - \frac{1}{r}\|\alpha A^T\lambda\|^2\\ &= \frac{1}{r}\|rx - \alpha A^T\lambda\|^2 + t\|\lambda\|^2 - \frac{1}{r}\|\alpha A^T\lambda\|^2.\end{aligned}$$

Since $r > 0$, we obtain

$$w^TQw \ge t\|\lambda\|^2 - \frac{1}{r}\|\alpha A^T\lambda\|^2. \tag{3.9}$$

Note that

$$\|\alpha A^T\lambda\|^2 = \alpha^2\lambda^T AA^T\lambda \le \alpha^2\rho(AA^T)\|\lambda\|^2,$$

we get

$$w^TQw \ge \left(t - \frac{1}{r}\alpha^2\rho(AA^T)\right)\|\lambda\|^2. \tag{3.10}$$

By (3.7) we have $w^TQw \ge 0$ and the assertion 1) is proven.

2). For $\forall\ w = (x, y, \lambda) \in W := X \times Y \times R^m$ and

$$Q = \begin{pmatrix} rI_{n_1} & 0 & -\alpha A^T\\ 0 & sI_{n_2} & -\alpha B^T\\ -\alpha A & -\alpha B & tI_m\end{pmatrix},$$

we have

$$\begin{aligned}wQw^T &= r\|x\|^2 - 2\alpha x^T(A^T\lambda) + s\|y\|^2 - 2\alpha y^T(B^T\lambda) + t\|\lambda\|^2\\ &= \frac{1}{r}\Big(\|rx\|^2 - 2\alpha(rx)^T(A^T\lambda) + \alpha^2\|A^T\lambda\|^2\Big)\\ &\quad + \frac{1}{s}\Big(\|sy\|^2 - 2\alpha(sy)^T(B^T\lambda) + \alpha^2\|B^T\lambda\|^2\Big) + t\lambda^T\lambda - \frac{\alpha^2}{r}\lambda^TAA^T\lambda - \frac{\alpha^2}{s}\lambda^TBB^T\lambda\\ &\ge \frac{1}{r}\|rx - \alpha A^T\lambda\|^2 + \frac{1}{s}\|sy - \alpha B^T\lambda\|^2 + \Big(t - \frac{\alpha^2\rho(AA^T)}{r} - \frac{\alpha^2\rho(BB^T)}{s}\Big)\|\lambda\|^2.\end{aligned}$$

Since $r, s > 0$, we obtain

$$\begin{aligned}wQw^T &\ge \left(t - \frac{\alpha^2\rho(AA^T)}{r} - \frac{\alpha^2\rho(BB^T)}{s}\right)\|\lambda\|^2\\ &\ge \left(\frac{t}{\beta_1} - \frac{\alpha^2\rho(AA^T)}{r}\right)\|\lambda\|^2 + \left(\frac{t}{\beta_2} - \frac{\alpha^2\rho(BB^T)}{s}\right)|\lambda\|^2.\end{aligned}$$

By the condition (3.8) we have $w^TQw \ge 0$ immediately and the assertion 2) is proven.

□

Notice that $0 < \alpha < 1$, which follows

1) $rt \ge \rho(AA^T) > \alpha^2\rho(A^TA)$,

2) $rt \ge \beta_1\rho(AA^T) > \beta_1\alpha^2\rho(AA^T)$, and $st \ge \beta_2\rho(BB^T) > \beta_2\alpha^2\rho(BB^T)$.

These imply that the parameters r, s and t are allowed to be chosen in so wide intervals that they are more likely to be proper for the specific situation. Moreover, the smaller r and s are, the closer the resolvent operators $(I + \frac{1}{r}\partial\theta_1)^{-1}$ and $(I + \frac{1}{s}\partial\theta_2)^{-1}$ are to $(\partial\theta_1)^{-1}$ and $(\partial\theta_2)^{-1}$ respectively, and the smaller t is, the larger the step-length of the dual iteration is. Thus, the proposed algorithms with $\alpha < 1$ could be fast. Without loss of generality, we take $\alpha \in (0, 1]$.

Theorem 3.2. *Let $\{w^k\}$ be a sequence satisfied the condition* (3.1) *and $w^* \in W^*$ be a solution defined by* (3.6)*. If F_α is monotone and the matrix Q is symmetric and positive semi-definite, then we have*

$$\|w^{k+1} - w^*\|_Q^2 \le \|w^k - w^*\|_Q^2 - \|w^k - w^{k+1}\|_Q^2. \tag{3.11}$$

Proof. By setting $w = w^*$ in (3.1), we get

$$\theta(u^*) - \theta(u^{k+1}) + (w^* - w^{k+1})^T \Big\{ F_\alpha(w^{k+1}) + Q(w^{k+1} - w^k) \Big\} \ge 0.$$

Hence

$$(w^* - w^{k+1})^T Q(w^{k+1} - w^k) \ge \theta(u^{k+1}) - \theta(u^*) - (w^* - w^{k+1})^T F_\alpha(w^{k+1}). \tag{3.12}$$

By the monotonicity of $F_\alpha(w)$, we get

$$(w^{k+1} - w^*)^T F_\alpha(w^{k+1}) \ge (w^{k+1} - w^*)^T F_\alpha(w^*). \tag{3.13}$$

Since $w^* \in W^*$ is a solution, it has

$$\theta(u^{k+1}) - \theta(u^*) + (w^{k+1} - w^*)^T F_\alpha(w^*) \ge 0. \tag{3.14}$$

Adding (3.13) and (3.14) yields

$$\theta(u^{k+1}) - \theta(u^*) + (w^{k+1} - w^*)^T F_\alpha(w^{k+1}) \ge 0. \tag{3.15}$$

Substituting (3.15) into (3.12), we get

$$(w^* - w^{k+1})^T Q(w^{k+1} - w^k) \ge 0. \tag{3.16}$$

Using the identity

$$(a-b)^T Q(c-d) = \frac{1}{2}\Big(\|a-d\|_Q^2 - \|a-c\|_Q^2\Big) + \frac{1}{2}\Big(\|c-b\|_Q^2 - \|d-b\|_Q^2\Big)$$

to the left side of (3.16), we have

$$0 \le (w^* - w^{k+1})^T Q(w^{k+1} - w^k) = \frac{1}{2}\Big(\|w^* - w^{\mathrm{k}}\|_Q^2 - \|w^* - w^{k+1}\|_Q^2 - \|w^k - w^{k+1}\|_Q^2\Big),$$

which implies (3.11) immediately and completes the proof. □

Theorem 3.3. *If a sequence $\{w^k\}$ satisfies the condition* (3.1)*, where Q is symmetric and positive semi-definite on W, then it is globally convergent to an element of the solution set W^* satisfying the condition* (3.6)*.*

Proof. Adding (3.11) with respect to k yields

$$\|w^{k+1} - w^*\|_Q^2 \le \|w^0 - w^*\|_Q^2 - \sum_{i=0}^{k} \|w^i - w^{i+1}\|_Q^2. \tag{3.17}$$

Since Q is positive semi-definite, we get

$$\|w^{k+1} - w^*\|_Q^2 \le \|w^0 - w^*\|_Q^2, \quad \forall\ k \ge 0,$$

which implies that the sequence $\{w^k\}$ is bounded. Moreover, we have

$$\sum_{i=0}^{k} \|w^i - w^{i+1}\|_Q^2 \le \|w^0 - w^*\|_Q^2 - \|w^{k+1} - w^*\|_Q^2,$$

which follows

$$\lim_{k\to\infty} \|w^k - w^{k+1}\|_Q^2 = 0. \tag{3.18}$$

Thus $\{w^k\}$ has at least one cluster point. Let $w^\infty := \lim_{k\to\infty} w^k$ be a cluster point of $\{w^k\}$. Taking limit on (3.1) and using (3.18), we have

$$\theta(u) - \theta(u^\infty) + (w - w^\infty)^T F_\alpha(w^\infty) \ge 0, \quad \forall\ w \in W, \tag{3.19}$$

which implies that $w^\infty \in W^*$ is a solution. □

In summary, if the parameters r and t satisfy the condition (3.7), then the sequence $\{w^k\}$ generated by the GCPPA globally converges to a solution of problem (2.1). Similarly, if the parameters r, s and t satisfy the condition (3.8), then the sequence $\{w^k\}$ generated by the ECPPA globally converges to a solution of problem (2.6).

So far, the global convergence of the GCPPA and ECPPA has been proven. In what follows, we will investigate the convergence rate of the proposed methods.

Lemma 3.1. *If the sequence $\{w^k\}$ satisfies condition* (3.1)*, where Q is a symmetric and positive semi-definite matrix. Then we have*

$$\|w^{k-1} - w^k\|_Q^2 \ge \|w^k - w^{k+1}\|_Q^2, \quad \forall\ k \ge 1. \tag{3.20}$$

Proof. Setting $w := w^k$ in (3.1), we get

$$\theta(u^k) - \theta(u^{k+1}) + (w^k - w^{k+1})^T \Big\{ F_\alpha(w^{k+1}) + Q(w^{k+1} - w^k) \Big\} \ge 0. \tag{3.21}$$

Since the VI (3.1) also holds at the $(k-1)^{th}$ iteration, we have

$$\theta(u) - \theta(u^k) + (w - w^k)^T \Big\{ F_\alpha(w^k) + Q(w^k - w^{k-1}) \Big\} \ge 0 \quad \forall\ w \in W.$$

Setting $w := w^{k+1}$ in the above inequality, we obtain

$$\theta(u^{k+1}) - \theta(u^k) + (w^{k+1} - w^k)^T \Big\{ F_\alpha(w^k) + Q(w^k - w^{k-1}) \Big\} \ge 0. \tag{3.22}$$

Adding (3.21) and (3.22) yields

$$(w^k - w^{k+1})^T Q \Big\{ (w^{k-1} - w^k) - (w^k - w^{k+1}) \Big\} \ge (w^{k+1} - w^k)^T (F_\alpha(w^{k+1}) - F_\alpha(w^k)).$$

By the monotonicity of F_α we obtain

$$(w^k - w^{k+1})^T Q\Big\{(w^{k-1} - w^k) - (w^k - w^{k+1})\Big\} \geq 0. \tag{3.23}$$

Adding $\big\|(w^{k-1} - w^k) - (w^k - w^{k+1})\big\|_Q^2$ to the both sides of (3.23), it follows

$$(w^{k-1} - w^k)^T Q\Big\{(w^{k-1} - w^k) - (w^k - w^{k+1})\Big\} \geq \Big\|(w^{k-1} - w^k) - (w^k - w^{k+1})\Big\|_Q^2.$$

By substituting $a = w^{k-1} - w^k$ and $b = w^k - w^{k+1}$ to the identity $\big\|a\big\|_Q^2 - \big\|b\big\|_Q^2 = 2a^T Q(a - b) - \big\|a - b\big\|_Q^2$, we obtain

$$\begin{aligned}
&\Big\|w^{k-1} - w^k\Big\|_Q^2 - \Big\|w^k - w^{k+1}\Big\|_Q^2 \\
=&2(w^{k-1} - w^k)^T Q\Big\{(w^{k-1} - w^k) - (w^k - w^{k+1})\Big\} - \Big\|(w^{k-1} - w^k) - (w^k - w^{k+1})\Big\|_Q^2 \\
\geq&\Big\|(w^{k-1} - w^k) - (w^k - w^{k+1})\Big\|_Q^2 \geq 0,
\end{aligned}$$

which follows (3.20) and completes the proof. □

Theorem 3.4. *Under the same conditions of Lemma 3.1, we have*

$$\|w^t - w^{t+1}\|_Q^2 \leq \frac{1}{t+1}\|w^0 - w^*\|_Q^2, \quad \forall\, w^* \in W^*. \tag{3.24}$$

Proof. By Lemma 3.1 and (3.17), we get

$$(t+1)\|w^t - w^{t+1}\|_Q^2 \leq \sum_{k=0}^{n} \|w^k - w^{k+1}\|_Q^2 \leq \|w^0 - w^*\|_Q^2,$$

which implies (3.24) immediately and completes the proof. □

Notice that the solution set W^* is a bounded and closed convex set (see Theorem 2.1 of He [21]). Let $d := \inf\Big\{\|w^0 - w^*\|_Q \;\Big|\; w^* \in W^*\Big\}$, then for any given $\varepsilon > 0$, the proposed methods need at most $k = \lfloor d^2/\varepsilon \rfloor$ iterations to ensure that $\|w^k - w^{k+1}\|_Q \leq \varepsilon$. Hence, the worst-case $O(1/k)$ convergence rate of the proposed methods is established.

4 Numerical results

The section focuses on the numerical performance of the GCPPA and ECPPA comparing with some existing state-of-the-art methods. The codes of all methods implemented are written in MATLAB 2010a and run on an Asus laptop computer with Intel(R) Core(TM) i3-2350M CPU 2.30GHz and 4G memory.

Example 4.1. The first test problem is the correlation matrices calibrating problem, which can be formulated as

$$\min\Big\{\, \frac{1}{2}\|X - C\|_F^2 \;\Big|\; \operatorname{diag}(X) = e,\ X \in S_+^n \Big\}, \tag{4.1}$$

where $e = (1, 1, \ldots, 1)^T \in \mathbb{R}^{n\times 1}$ and $S_+^n = \{H \in \mathbb{R}^{n\times n} \mid H^T = H, H \succeq 0\}$. Problem (4.1) can be reformulated to the form of (1.1) with $\rho(A^T A) = 1$.

We compare the performance of the GCPPA with the CPPA and RCPPA. In the experiment, the matrix C is given in a random style:

$$C = \text{rand}(n, n), \quad C = (C' + C) - \text{ones}(n, n).$$

For each given n, 20 random instances are tested. To be fair, the initial point of all methods is set to $x^0 = eye(n, n), \lambda^0 = zeros(n, 1)$, and the termination criterion of all methods is set to $\max\{\|w^{k+1} - w^k\|\} \leq 10^{-5}$. Set $r = 2.00$ and $t = \frac{1.05}{2}$, which satisfy $rt \geq \rho(A^T A) = 1$, and set $\gamma = 1.50$ in the CPPA and RCPPA. Set $\alpha = 0.20$, $r = 0.6$ and $t = 0.7$ which satisfy $rt = 1.05\alpha^2 \geq \alpha^2 \rho(A^T A) = 1 \times \alpha^2$ in the GCPPA. For easy comparisons, the results of the three methods in terms of matrix dimension n, the average iteration number (iter) and the average cpu-time (cput, in seconds) are put into Table 4.1, and displayed in Figure 4.1.

Table 4.1: Numerical results of CPPA, RCPPA and GCPPA (cput in seconds)

n	CPPA		RCPPA		GCPPA	
	iter	cput	iter	cput	iter	cput
100	31	0.56	23	0.38	19	0.31
200	34	2.48	25	1.75	21	1.51
300	37	6.43	28	4.69	22	3.83
400	38	12.96	27	8.88	23	7.97
500	40	27.54	28	20.44	24	18.97
600	41	44.80	28	33.69	24	29.13
700	42	80.20	29	54.92	26	51.44
800	43	104.81	30	72.99	26	64.71
900	44	142.55	30	94.31	30	98.14
1000	51	244.05	32	168.69	32	165.32

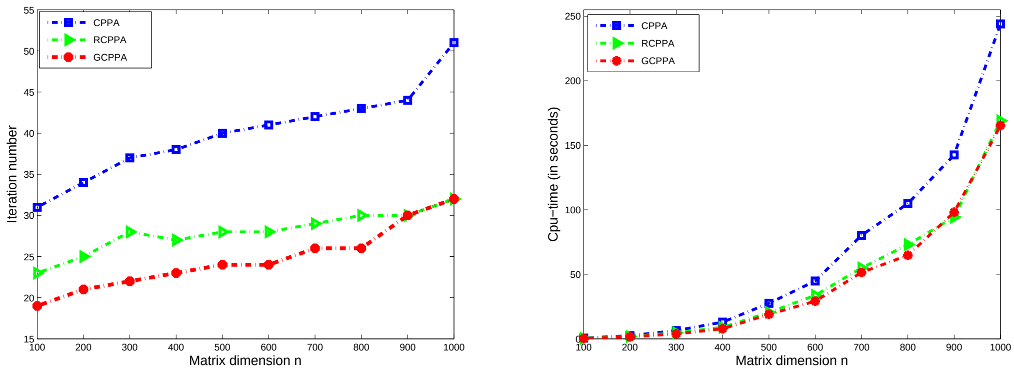

Figure 4.1: Comparisons of CPPA, RCPPA and GCPPA. Left: iteration number. Right: cputime.

One can conclude from Table 4.1 and Figure 4.1 that, the GCPPA takes about 60% iterations of the CPPA, and takes less cpu-time than the CPPA especially for the high dimension problem. The GCPPA has almost the same numerical performance as the RCPPA with $\gamma = 1.50$.

Example 4.2. The second test problem is the matrix completion problem utilized by Tao, Yuan and He [23]

$$\min\left\{\frac{1}{2}\|X-C\|_F^2 \;\middle|\; X\in S_+^n\cap S_B\right\}, \tag{4.2}$$

where $S_+^n=\{H\in\mathbb{R}^{n\times n}\mid H^T=H, H\succeq 0\}$ and $S_B=\{H\in\mathbb{R}^{n\times n}\mid H_L\geq H\geq H_U\}$.

By introducing an equality constraint $X-Y=0$, problem (4.2) is equivalent to the separable convex optimization problem of the form

$$\min\left\{\frac{1}{2}\|X-C\|_F^2+\frac{1}{2}\|Y-C\|_F^2 \;\middle|\; X-Y=0,\ X\in S_+^n,\ Y\in S_B\right\}. \tag{4.3}$$

Obviously, problem (4.3) is a special case of problem (1.7) with $\rho(A^TA)=1$ and $\rho(B^TB)=1$. For comparison, the following four methods are used to solve the test problem (4.3): classical ADMM (ADMM for short), GADMM, separable version of CPPA (i.e., Algorithm 2.2 with $\alpha=1$, SCPPA for short) and ECPPA (Algorithm 2.2 with $0<\alpha_0<\alpha<1$).

In the numerical experiment, we take the matrix C in the random style

$$C=\text{rand}(n,n),\quad C=(C^T+C)-\text{ones}(n,n).$$

To be fair, we set the initial point of all used methods to

$$x^0=I_{n\times n},\quad y^0=-I_{n\times n},\quad \lambda^0=0_{n\times n}.$$

The termination criterion of all methods is set to

$$\frac{\|w^{k+1}-w^k\|_\infty}{\|w^1-w^0\|_\infty}\leq 10^{-5}.$$

The parameter settings of the used methods are stated as follows. The penalty parameter is set to $\beta=1.80$ in both ADMM and GADMM, the relaxation parameter is set to $\gamma=1.8$ in the GADMM. Let $R=3$, the proximal parameters of the SCPPA are set to $r=s=R$ and $t=\frac{1.02}{R}$, and those of the ECPPA are set to $r=\frac{3}{2}\alpha\beta_1\times R$, $s=\frac{3}{2}\alpha\beta_2\times R$, $t=\frac{2}{3}\alpha\times\frac{1.02}{R}$, where $\alpha=0.34$ and $\beta_1=\beta_2=2$. It is obvious that the condition (3.8) is satisfied by this settings.

Table 4.2: Numerical results of ADMM, GADMM, SCPPA and ECPPA (cput in seconds)

n	ADMM		GADMM		SCPPA		ECPPA	
	iter	cput	iter	cput	iter	cput	iter	cput
100	120	1.86	60	0.86	57	0.84	40	0.58
200	127	14.04	64	7.10	60	6.30	41	4.32
300	146	57.34	74	31.08	65	23.89	40	14.37
400	145	61.65	73	30.74	68	28.28	42	19.30
500	153	123.71	77	62.85	70	53.60	45	35.34
600	169	283.06	84	141.91	76	125.61	50	91.06
700	179	427.06	89	213.53	86	197.87	55	130.35
800	162	431.49	80	217.71	87	234.48	56	155.64
900	180	901.00	89	446.12	99	477.23	64	322.01
1000	183	1066.22	90	523.55	101	583.54	69	401.29

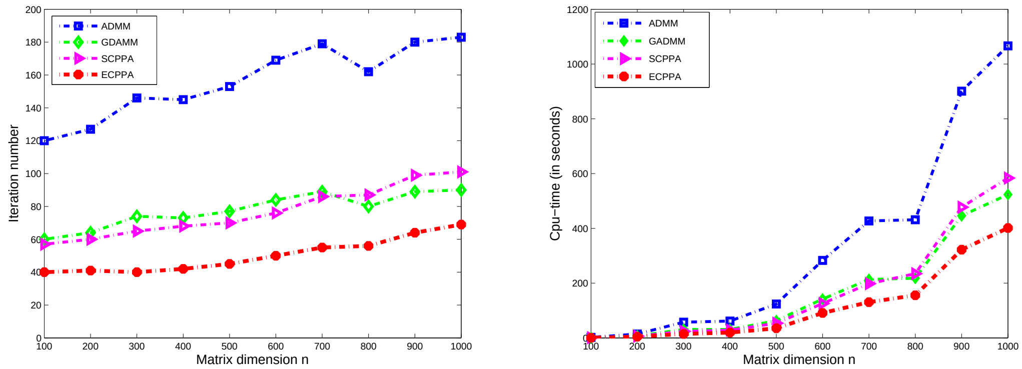

Figure 4.2: Comparisons of ADMM, GADMM, SCPPA and ECPPA. Left: iteration number. Right: cpu time.

We test the methods on 20 random instances for each fixed matrix dimension n. The average iteration number and the average cpu-time of the methods for problem (4.3) with different matrix dimension n are listed in Table 4.2 and displayed in Figure 4.2.

Observing from Table 4.2 and Figure 4.2, one can conclude that, the iteration number of ECPPA is about 30% of ADMM, and about 60% of GADMM and SCPPA. Moreover, ECPPA takes less cpu-time than the others, especially for the high dimensional problem. Thus, ECPPA is more efficient than the other three methods.

Example 4.3. This example focuses on the wavelet-based image inpainting and zooming problems. Let $\mathbf{x} \in R^l$ represent an $l_1 \times l_2$ image with $l = l_1 \cdot l_2$ (the two-dimensional images are tackled by vectorizing them as one-dimensional vectors, e.g., in the lexicographic order), and let $W \in R^{l\times n}$ be a wavelet dictionary, i.e., each column of W be the elements of a wavelet frame. Commonly, the image $\mathbf{x}$ possesses a sparse representation under the dictionary W , i.e., $\mathbf{x} = Wx$ with x being a sparse vector. The wavelet-based image processing thus considers recovering the real image $\mathbf{x}$ from an observation b which might have some missing pixels or convolutions. The model for the wavelet-based image processing can be casted as

$$\min\{\|x\|_1 \Big| BWx = b\}, \tag{4.4}$$

where $\|x\|_1$ is to deduce a sparse representation under the wavelet dictionary; and B (also called mask) is a typically ill-conditioned matrix representation of convolution or downsampling operators. For the inpainting problem, the matrix $B \in R^{l\times l}$ in (4.4) is a diagonal matrix whose diagonal elements are either 0 or 1, where the locations of 0 correspond to missing pixels in the image and locations of 1 correspond to the pixels to be kept. For the image zooming problem, the matrix $B \in R^{m\times l}$ can be expressed as $B = SH$ where $S \in R^{m\times l}$ is a downsampling matrix and $H \in R^{l\times l}$ is a blurry matrix, where H can be diagonalized by the discrete consine transform (DCT), i.e., $H = C^{-1}\Lambda C$ where C represents the DCT and Λ is a diagonal matrix whose diagonal entries are eigenvalues of H.

We test the algorithms on the 256×256 images of Peppers.png and Boat.png for the image inpainting and image zooming problems, respectively. Both the clean and degraded images are displayed in Fig.4.3. The dictionary W is chosen as the inverse discrete Haar wavelet transform

Figure 4.3: Original Peppers, degraded Peppers, original Boat, degraded Boat

with a level of 6. Below we give the detail of how the tested images are degraded.

- For the image inpainting problem, the original image Peppers is first blurred by the out-of-focus kernel with a radius of 7. Then 60% pixels of the blurred images are lost by implementing a mask operator S. The positions of missing pixels are located randomly.
- For the image zooming problem, the original image Boat is downsampled by a downsampling matrix S with a factor of 4. Then, the downsampled image is corrupted by a convolution whose kernel is generated by `fspecial(gaussian,9,2.5)` of MATLAB.

On the above problems, the performance of the GCPPA (compared with CPPA and RCPPA) is tested. Since the dictionary W has the property $WW^T = I$, the blurry matrix H can be diagonalized by DCT and the binary matrix (both mask and downsampling matrices) S satisfies $\|S\| = 1$, we have $\|A^T A\| = 1$ (where $A := BW$) for the wavelet-based image inpainting and zooming problems. Therefore, the requirement $rs > \|A^T A\|$ reduces to $rs > 1$ for CPPA or RCPPA, and $rs > \alpha^2$ for GCPPA. The implementation details about how to choose the involved parameters are described as below.

- For the image inpainting problem, we take the initial point $(x^0, \lambda^0) = (W^T(b), \mathbf{0})$, and $r_0 = 0.6$, $s_0 = \frac{1.02}{r_0}$, $\gamma = 1$ for CPPA and $\gamma = 1.9$ for RCPPA; $a = 0.5$, $r_1 = \frac{5}{2}\alpha \times r_0$, $s_1 = \frac{2}{5}\alpha \times \frac{1.02}{r_0}$ for GCPPA;
- For the image zooming problem, we take the initial point $(x^0, \lambda^0) = (\mathbf{0}, \mathbf{0})$, and $r_0 = 0.55$, $s_0 = \frac{1.02}{r_0}$, $\gamma = 1$ for CPPA and $\gamma = 1.2$ for RCPPA; $a = 0.5$, $r_1 = \frac{5}{2}\alpha \times r_0$, $s_1 = \frac{2}{5}\alpha \times \frac{1.02}{r_0}$ for GCPPA;

As usual, the quality of the reconstruction is measured by the signal-to-noise ratio (SNR) in decibel (dB)

$$SNR := 20 \log_{10} \frac{\|\mathbf{x}\|}{\|\bar{\mathbf{x}} - \mathbf{x}\|}, \tag{4.5}$$

where $\bar{\mathbf{x}}$ is a reconstructed image and $\mathbf{x}$ is a clean image.

In Fig.4.4, we plot the evolutions of SNR with respect to iterations and computing time for all the tested algorithms. It shows that RCPPA is better than CPPA, and GCPPA converges to the nearly same solutions in inpainting and zooming problems as fast as RCPPA. Then, we

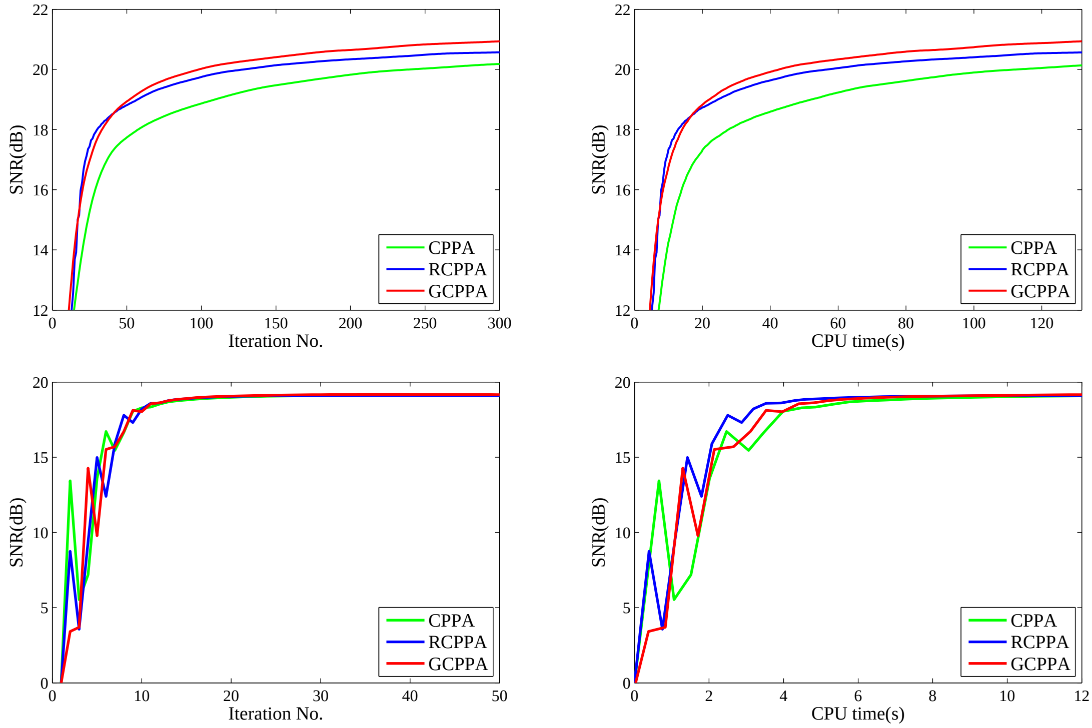

Figure 4.4: Evolutions of SNR with respect to the number of iterations (Iteration No.) and computing time (CPU time) (*Up row:* for image inpainting; *Bottom row:* for image zooming).

display the reconstructed images in Fig.4.5 by executing 300 iterations. It verifies our assertion: the GCPPA could be at least as efficient as the RCPPA on the inpainting and zooming problem. We have to emphasize that, the GCPPA does not involve any relaxation step while the RCPPA has a relaxation step with $\gamma = 1.5 > 1$.

Example 4.4. This example focuses on the total variation (TV) uniform noise removal model:

$$\min\{\|\nabla \mathbf{x}\|_1 \Big| \|H\mathbf{x} - \mathbf{x}^0\|_\infty \leq \sigma\}, \tag{4.6}$$

where $\mathbf{x}^0 \in R^l$ is an observed image corrupted by a zero-mean uniform noise; $H \in R^{l\times l}$ is the matrix representation of a blurry operator as in Example 4.3, $\|\nabla \cdot\|_1$ is the TV norm (see e.g., [22]), σ is a parameter indicating the uniform noise level and $\|\mathbf{x}\|_\infty := \max_{1\leq i\leq l} \|\mathbf{x}_i\|$.

By denoting

$$A := \begin{pmatrix} H \\ -H \end{pmatrix}, \quad \textit{and} \quad b := \begin{pmatrix} \mathbf{x}^0 - \sigma\mathbf{e} \\ -\mathbf{x}^0 - \sigma\mathbf{e} \end{pmatrix},$$

with $\mathbf{e} = (1, 1, ..., 1)^T \in R^l$, the model (4.6) amounts to

$$\min\{\|\nabla \mathbf{x}\|_1 | A\mathbf{x} \leq b\}, \tag{4.7}$$

We also test the algorithms on the 256-by-256 images of Peppers.png and Boat.png. The clean images are degraded by either the Gaussian (`fspecial(gaussian,9,2.5)`) or the out-of-focus (`fspecial(disk,3)`) convolution. Then, the degraded images are further corrupted by the

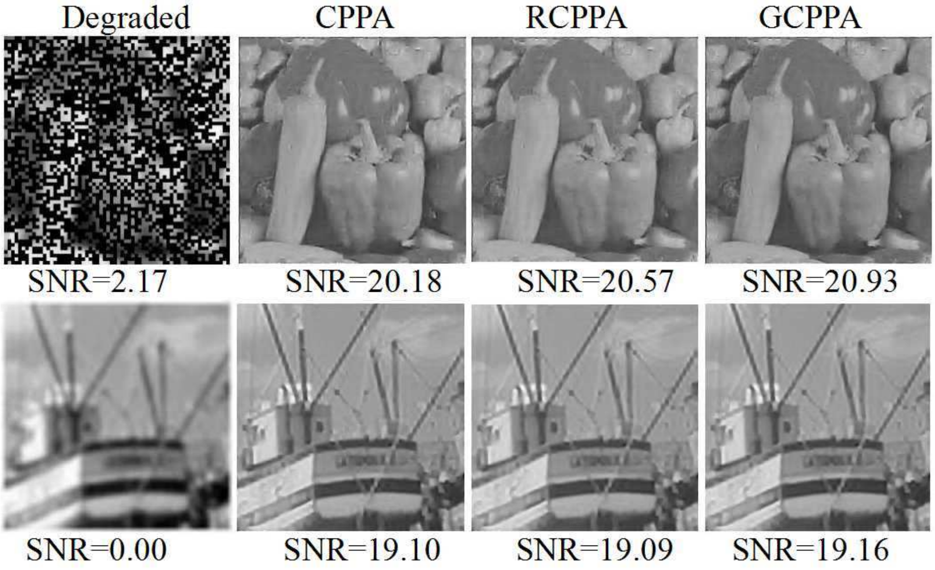

Figure 4.5: Reconstructed images by CPPA, RCPPA and GCPPA (*Up row:* for image inpainting; *Bottom row:* for image zooming).

zero-mean uniform noise with $\sigma = 0.2$ or 0.5. We take $r = 0.6$, $s = 1.02/r$ and $\gamma = 1.8$ for CPPA and RCPPA; $a = 0.5$, $r_1 = \frac{5}{2}\alpha \times r_0$, $s_1 = \frac{2}{5}\alpha \times \frac{1.02}{r_0}$ for GCPPA. The initial points for all the tested algorithms are taken as zeros and the stopping criterion is set to

$$Tol = \frac{\|\mathbf{x}^{k+1} - \mathbf{x}^k\|}{\|\mathbf{x}^k\|} < 10^{-6}. \tag{4.8}$$

For the TV uniform noise removal model (4.7), the iteration scheme of CPPA or RCPPA can be decribed as

$$\begin{cases} \tilde{\lambda}^k = P_{R_+}\Big\{\lambda^k - \dfrac{1}{s}(A\mathbf{x}^k - b)\Big\}, \\ \tilde{\mathbf{x}}^k = \arg\min\Big\{\|\nabla\mathbf{x}\|_1 + \dfrac{r}{2}\|\mathbf{x} - \mathbf{x}^k - \dfrac{1}{r}(2\tilde{\lambda}^k - \lambda^k)\|^2 \Big| \mathbf{x} \in X\Big\}, \end{cases} \tag{4.9}$$

and the relaxation step

$$(\mathbf{x}^{k+1}, \lambda^{k+1}) = (\mathbf{x}^k, \lambda^k) - \gamma\Big((\mathbf{x}^k, \lambda^k) - (\tilde{\mathbf{x}}^k, \tilde{\lambda}^k)\Big) \tag{4.10}$$

is used. The GCPPA has the different scheme decribed as

$$\begin{cases} \lambda^{k+1} = P_{R_+}\Big\{\lambda^k - \dfrac{\alpha}{s}(A\mathbf{x}^k - b)\Big\}, \\ \mathbf{x}^{k+1} = \arg\min\Big\{\|\nabla\mathbf{x}\|_1 + \dfrac{r}{2}\|\mathbf{x} - \mathbf{x}^k - \dfrac{1}{r}[(\alpha+1)\lambda^{k+1} - \alpha\lambda^k]\|^2 \Big| \mathbf{x} \in X\Big\}. \end{cases} \tag{4.11}$$

The numerical results are put in Table 4.3 for comparison. Each set of $\cdot \backslash \cdot \backslash \cdot$ represents the number of iterations, the computing time in seconds and the restored SNR when the stopping

Table 4.3: Numerical results of CPPA, RCPPA and GCPPA

Blur	σ	Images	CPPA	RCPPA	GCPPA
Gaussian	0.2	Peppers	72/16.1/16.75	87/20.0/17.13	61/14.5/17.40
		Boat	65/15.1/16.58	79/18.4/17.01	58/13.1/17.31
	0.5	Peppers	103/22.3/16.00	121/27.2/16.38	71/16.6/16.58
		Boat	83/19.6/15.85	101/24.8/16.33	72/16.3/16.61
Out-of-focus	0.2	Peppers	71/16.8/17.67	88/19.3/18.12	53/11.9/18.36
		Boat	65/16.4/17.53	82/20.7/18.07	54/12.8/18.37
	0.5	Peppers	94/22.0/16.65	108/24.7/17.01	65/15.5/17.17
		Boat	86/21.6/16.57	104/24.1/17.05	64/14.7/17.25

criterion (4.8) is reached. Observing from table 4.3, we can conclude that: RCPPA has better performance comparing with CPPA in SNR but worse in the number of iterations and cpu-time, while GCPPA has better performance comparing with CPPA in all terms including SNR, the number of iterations and cpu-time.

The images Peppers.png and Boat.png are degraded with the Gaussian or the out-of-focus convolution. The degraded images and the restored images by all the tested algorithms are shown in Figures 4.6 and 4.7, respectively.

5 Conclusions

In this paper, a new customized proximal point algorithm, GCPPA, is proposed for convex optimization problem. The new method is essentially a generalization of the customized proximal point algorithm (CPPA) proposed by He, Yuan and Zhang [10]. The primal iteration subproblem of CPPA is

$$\tilde{x}^k = \arg\min\Big\{\theta(x) + \frac{r}{2}\big\|(x - x^k) - \frac{1}{r}A^T(2\tilde{\lambda}^k - \lambda^k)\big\|^2 \Big| x \in X\Big\}, \tag{5.1}$$

and

$$(x^{k+1}, \lambda^{k+1}) = (x^k, \lambda^k) - \gamma\Big((x^k, \lambda^k) - (\tilde{x}^k, \tilde{\lambda}^k)\Big). \tag{5.2}$$

While the primal iteration subproblem of the proposed GCPPA is

$$x^{k+1} = \arg\min\Big\{\theta(x) + \frac{r}{2}\big\|(x - x^k) - \frac{1}{r}A^T[(1+\alpha)\lambda^{k+1} - \alpha\lambda^k]\big\|^2 \Big| x \in X\Big\}, \tag{5.3}$$

where $0 < \alpha \le 1$. The CPPA with $\gamma = 1$ is a special case of the GCPPA with $\alpha = 1$. A superiority of GCPPA is that, the CPPA has to take a relaxation step (i.e. $\gamma > 1$) for fast convergence, while GCPPA does not involve any relaxation step and has better performance comparing with CPPA. Hence, the GCPPA is more suitable to some real applications in which the relaxation step is not permitted, such as some real-time control systems and game theory models.

In this paper, the GCPPA is further used to the separable convex optimization problem with linear constraints, and the extended version is named as ECPPA for short. Global convergence and the worst-case $O(1/k)$-convergence rate of both GCPPA and ECPPA are proved in a uniform framework. Some numerical experiments have shown that:

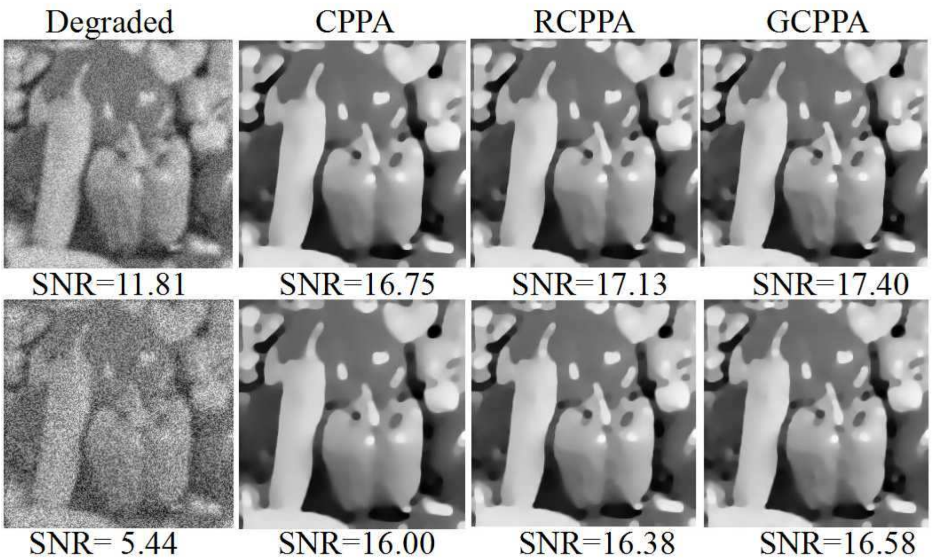

Figure 4.6: Degraded image by the Gaussian convolution and restored image(*Up row:* corrupted by $\sigma = 0.2$; *Bottom row:* corrupted by $\sigma = 0.5$)

1) For the correlation matrices calibrating problem, GCPPA outperforms CPPA with (or without) the relaxation step.

2) For the matrix completion problem, the ECPPA has higher efficiency comparing with some existing state-of-the-art methods such as ADMM, GADMM and a separable-version of CPPA.

3) For inpainting and zooming problems, GCPPA outperforms the CPPA without relaxation step, and it could be at least as efficient as the CPPA with the relaxation step.

4) For the TV uniform noise removal model, GCPPA obtains better restored images with less cpu-time and fewer iterations than the CPPA with (or without) the relaxation step.

In summary, the proposed algorithms (GCPPA and ECPPA) in this paper do not involve any relaxation step while they still keep the same convergence properties as the CPPA with some relaxation steps. Numerical results have demonstrated that the proposed methods are more effective and high efficiency than some existing state-of-the-art methods.

Acknowledgments The authors are very grateful to Doctor W-X Zhang at University of Electronic Science and Technology of China for his help on numerical experiments on image processing problems.

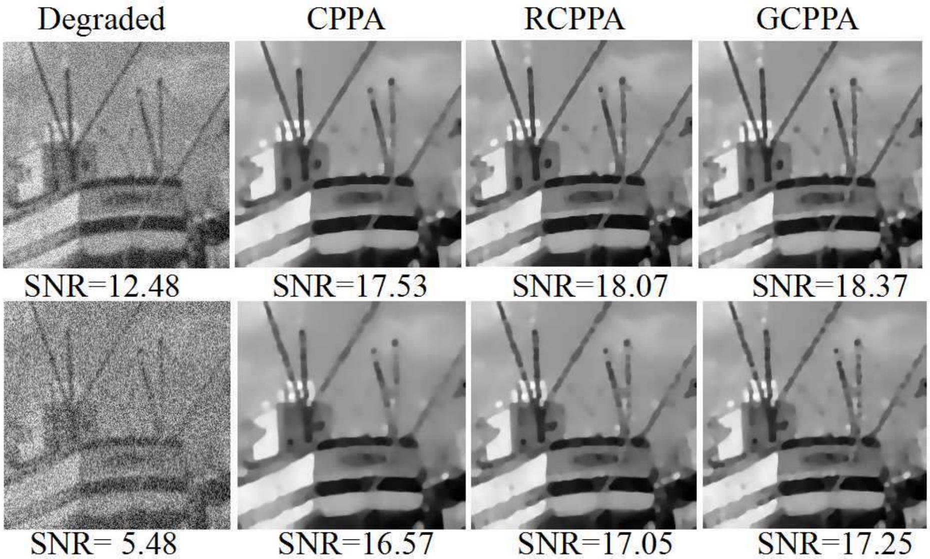

Figure 4.7: Degraded image by the out-of-focus convolution and restored image (*Up row:* corrupted by $\sigma = 0.2$; *Bottom row:* corrupted by $\sigma = 0.5$)

References

[1] J.J. Moreau, Proximité et dualité dans un espace hilbertien, Bull. Soc. Math. France, 93, 273-299 (1965)

[2] R.T. Rockafellar, Monotone operators and the proximal point algorithm, SIAM J. Control Optim., 14(5), 877-898 (1976)

[3] D.P. Bertsekas, P. Tseng, Partial proximal minimization algorithms for convex programming, SIAM J. Control Optim., 4(3), 551-572 (1993)

[4] A. Kaplan, R. Tichatschke, Proximal point methods and nonconvex optimization, J. Global Optim., 13(4), 389-406 (1998)

[5] B. Martinet, Regularisation, d'iné quations variationelles par approximations succesives, Rev. France Informat. Rech. Oper., 4, 154-159 (1970)

[6] G. Kassay, The proximal points algorithm for reflexive Banach spaces, Studia Math., 30, 9 -17 (1985)

[7] O. Güler, New proximal point algorithms for convex minimization, SIAM J. Optim., 2(4), 646-664 (1992)

[8] S. Grotzinger, C. Witzgall, Partial inverse of a monotone operator, Appl. Math. Optim., 10(10), 247-265 (1983)

[9] S.P. Han, A decomposition method and its application to convex programming, Math. Oper. Res., 14(2), 237-248 (1989)

[10] B.S. He, X.M. Yuan, W. Zhang, A customized proximal point algorithm for convex minimization with linear constraints, Comput. Optim. Appl., 56(3), 559-572 (2013)

[11] M.R. Hestenes, Multiplier and gradient methods, J. Optim. Theory Appl., 4(4), 303-320 (1969)

[12] M.J.D. Powell, A method for nonlinear constraints in minimization problems, Optimization, 5(6), 283-298 (1969)

[13] R.T. Rockafellar, Augmented Lagrangians and applications of the proximal point algorithm in convex programming, Math. Oper. Res., 1(2), 97-116 (1976)

[14] B.S. He, Parallel splitting augmented Lagrangian methods for monotone structured variational inequalities, Comput. Optim. Appl., 42(2), 195-212 (2009)

[15] R. Glowinski, A. Marrocco, Analyse numerique du champ magnetique d'un alternateur par elements finis et sur-relaxation ponctuelle non lineaire, Comput. Methods Appl. Mech. Engin., 3(1), 55-85 (1974)

[16] D. Gabay, B. Mercier, A dual algorithm for the solution of nonlinear variational problems via finite element approximation, Comput. Math. Appl., 2(1), 17-40 (1976)

[17] D. Gabay, Applications of the method of multipliers to variational inequalities, M. Fortin, R. Glowinski (eds.), Augmented Lagrangian Methods: Applications to the Numerical Solution of Boundary-Value Problems, 299-331. North-Holland, Amsterdam (1983)

[18] J. Douglas, H.H. Rachford, On the numerical solution of heat conduction problems in two and three space variables, Trans. Amer. Math. Soc., 82(2), 421- 439 (1956)

[19] J. Eckstein, D. P. Bertsekas, On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators, Math. Program., 55(3), 293-318 (1992)

[20] B.S. He, L.Z. Liao, M.J. Qian, Alternating projection based prediction-correction methods for structured variational inequalities, J. Comput. Math., 24(6), 693-710 (2006)

[21] B.S. He, X.M. Yuan, On the $O(1/n)$ convergence rate of the Douglas-Rachford alternating direction method, SIAM J. Numer. Anal., 50(2), 700-709 (2012)

[22] B.S. He, X.M. Yuan, On non-ergodic convergence rate of Douglas-Rachford alternating direction method of multipliers, Numer. Math., 130(3), 567-577 (2014)

[23] M. Tao, X.M. Yuan, B.S. He, Solving a class of matrix minimization problems by linear variational inequality approaches, Linear Algebra Appl., 434(11), 2343-2352 (2011)