A relaxed customized proximal point algorithm for separable convex programming

Xingju Cai [*†] Guoyong Gu[*‡] Bingsheng He[*§] Xiaoming Yuan [¶]

August 22, 2011

Abstract

The alternating direction method (ADM) is classical for solving a linearly constrained separable convex programming problem (primal problem), and it is well known that ADM is essentially the application of a concrete form of the proximal point algorithm (PPA) (more precisely, the Douglas-Rachford splitting method) to the corresponding dual problem. This paper shows that an efficient method competitive to ADM can be easily derived by applying PPA directly to the primal problem. More specifically, if the proximal parameters are chosen judiciously according to the separable structure of the primal problem, the resulting customized PPA takes a similar decomposition algorithmic framework as that of ADM. The customized PPA and ADM are equally effective to exploit the separable structure of the primal problem, equally efficient in numerical senses and equally easy to implement. Moreover, the customized PPA is ready to be accelerated by an over-relaxation step, yielding a relaxed customized PPA for the primal problem. We verify numerically the competitive efficiency of the customized PPA to ADM, and the effectiveness of the over-relaxation step. Furthermore, we provide a simple proof for the $O(1/t)$ convergence rate of the relaxed customized PPA.

Keywords: Proximal point algorithm, convex programming, separable structure, alternating direction method, convergence rate, over-relaxation

1 Introduction

Both the augmented Lagrangian method (ALM, [15, 25]) and the proximal point algorithm (PPA, [21, 28] play fundamental roles in the area of optimization, and they are related to each other in many ways. A perfect illustration (see [27]) is that when a linearly constrained convex programming problem (the primal problem) is considered

$$\min\{\theta(u) \mid Cu = b,\ u \in \mathcal{U}\}, \tag{1.1}$$

where $\theta(u) : \mathbb{R}^n \to \mathbb{R}$ is a closed convex functions (not necessarily smooth), $C \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$ and $\mathcal{U} \subseteq \mathbb{R}^n$ is a closed convex set; the ALM for (1.1) is essentially the application of PPA to the dual problem of (1.1).

[*]Department of Mathematics, Nanjing University, Nanjing, 210093, China.

[†]Email: caixingju@njnu.edu.cn

[‡]Email: ggu@nju.edu.cn

[§]This author was supported by the NSFC Grant 10971095 and the NSF of Jiangsu Province Grant BK2008255. Email: hebma@nju.edu.cn

[¶]Corresponding author. Department of Mathematics, Hong Kong Baptist University, Hong Kong, China. This author was supported by an Hong Kong General Research Fund. Email: xmyuan@hkbu.edu.hk

The primal-dual relationship between ALM and PPA for (1.1) takes a splitting illustration when a separable form of (1.1) is considered. More specifically, we consider

$$\min\{F(x) + G(y) \mid Ax + By = b,\ x \in \mathcal{X}, y \in \mathcal{Y}\}, \tag{1.2}$$

where $F(x) : \mathbb{R}^{n_1} \to \mathbb{R}$, $G(x) : \mathbb{R}^{n_2} \to \mathbb{R}$ are closed convex functions (not necessarily smooth), $A \in \mathbb{R}^{m\times n_1}$, $B \in \mathbb{R}^{m\times n_2}$, $b \in \mathbb{R}^m$, $\mathcal{X} \subseteq \mathbb{R}^{n_1}$, $\mathcal{Y} \subseteq \mathbb{R}^{n_2}$ are closed convex sets; and $n_1 + n_2 = n$. Throughout the paper we assume that the solution set of (1.2) is nonempty. Then, it was elaborated in [9] that the Douglas-Rachford splitting method [8] which was well known previously in the area of partial differential equation is actually a special splitting form of PPA; and when this particular splitting form of PPA is applied to the dual problem of (1.2), the alternating direction method (ADM) in [12] which is a splitting version of ALM customized for (1.2) is recovered. Overall, for either the generic linearly constrained convex programming (1.1) or the separable form (1.2), PPA plays a crucial role in inspiring efficient structure-exploited algorithms in the literature. But, as far as we know, the central role of PPA has only been highlighted in the context of dual problems, and the direct application of PPA to the primal problems (1.1) or (1.2) is in infancy in the literature.

In [17], we studied the direct application of PPA to the generic form (1.1), showing the possibility of deriving an implementable PPA which handles the primal and dual variables individually. Considering the wide applications of the separable model (1.2) in various areas such as image processing and statistical learning, it is urged to continue along this line of idea and to further consider the direct application of PPA to (1.2). This is the main purpose of the paper. No doubt the separable structure of (1.2) sheds the light on developing a customized PPA which can exploit the particular structure of (1.2) fully. Our objective of the algorithmic design is that the PPA subproblems should be also in a decomposed form where the primal variables x and y are handled separably by exploiting the properties of $F(x)$ and $G(y)$ individually, exactly as what ADM does. This desire comes from the impressively wide applicability and numerical efficiency of ADM which have been witnessed ever since its presence in [12], and especially in recent literature. Therefore, by determining appropriate proximal parameters according to the separable structure of (1.2), a customized PPA is developed for (1.2). As we shall show, the subproblems of the customized PPA are as decomposable as those of ADM, with the only difference in the order of updating the primal variable y and the dual variable. Therefore, the customized PPA and ADM are equally effective to exploit the separable structure of the primal problem (1.2) and equally easy to implement. We thus expect that they are equally efficient in numerical senses, which will be verified by preliminary numerical experiments.

In addition, we follow the relaxed PPA promoted by Gol'shtein and Tret'yakov in [13], and accordingly develop a relaxed customized PPA which combines the customized PPA with an over- or under-relaxation step. By numerical evidence, we shall demonstrate that an over-relaxation step can accelerate the customized PPA easily without additional cost. In the literature, due to the simplicity and efficiency of ADM, it has been a long effort to investigate how to accelerate ADM. For example, the existing effort in [35] attempts to accelerate ADM via a descent step. This descent step, however, requires demanding computation for identifying an appropriate step size at each iteration, and this difficulty excludes the application of accelerated ADMs in some interesting applications such as image processing and statistics (see e.g.[16]). The relaxation step of the relaxed PPA in [13], however, is free from computing any step size, and it is simply a linear combination with a constant coefficient. Therefore, by regarding the customized PPA as a practical surrogate of the ADM, the relaxed customized PPA can be viewed as the first efficient realization of the acceleration of ADM in the literature. Our theoretical contribution is that we will prove that the relaxed customized PPA enjoys the same $O(1/t)$ convergence rate as that of ADM (which was proved in [18]). As a by-product, the proof of the $O(1/t)$ convergence rate provides a theoretical explanation on the conjectured assertion posed in [3]: it is preferred to choose an over-relaxation factor of the relaxed PPA in the interval $(1, 2)$ empirically.

Overall, the contribution of this paper can be summarized as follows:

1). A demonstration that a customized application of PPA to the primal problem (1.2) can be as efficient as that to its dual problem (i.e., ADM), and thus the customized PPA can be viewed as a practical surrogate of ADM in the primal context;

2). A relaxed customized PPA is developed for (1.2) which accelerates the customized PPA easily with very cheap cost;

3). The $O(1/t)$ convergence rate of the relaxed customized PPA is proved.

The rest of the paper is organized as follows. In Section 2, we briefly review some preparatory knowledge which is useful for further discussions. In Section 3, we propose the relaxed customized PPA and elaborate on the motivation. Then, we prove the global convergence of the new method in Section 4 and the $O(1/t)$ convergence rate in Section 5. In Section 6, we show the numerical comparison of the new method with some existing methods. The acceleration contributed by the over-relaxation factor is also verified in this section. Finally, some concluding remarks are given in Section 7.

2 Preliminaries

In this section, we provide some preliminaries which are useful for further discussions. In particular, we review a variational reformulation of (1.2), the relaxed PPA in [13] for solving the variational reformulation, and the ADM in [12] for solving (1.2) directly.

We first show that the model (1.2) can be characterized by a variational reformulation. By attaching a Lagrange multiplier $\lambda \in \mathbb{R}^m$ to the linear constraint, the Lagrange function of (1.2) is

$$L(x, y, \lambda) = F(x) + G(y) - \lambda^T(Ax + By - b), \tag{2.1}$$

and it is defined on the set

$$\Omega := \mathcal{X} \times \mathcal{Y} \times \mathbb{R}^m.$$

Let $\big(x^*, y^*, \lambda^*\big)$ be a saddle point of the Lagrange function (2.1). Then, for any $(x, y, \lambda) \in \Omega$, we have

$$L(x^*, y^*, \lambda) \le L(x^*, y^*, \lambda^*) \le L(x, y, \lambda^*).$$

Let $\partial F(x)$ and $\partial G(y)$ be the subdifferential operators of the convex functions $F(x)$ and $G(y)$, respectively. We use $f(x) \in \partial F(x)$ and $g(y) \in \partial G(y)$ to denote a subgradient of $F(x)$ and $G(y)$, respectively. Then, finding a saddle point of $L(x, y, \lambda)$ is equivalent to finding $w^* = (x^*, y^*, \lambda^*) \in \Omega$, $f(x^*) \in \partial F(x^*)$ and $g(y^*) \in \partial G(y^*)$ such that the following inequalities

$$\begin{cases} (x - x^*)^T(f(x^*) - A^T\lambda^*) \ge 0, \\ (y - y^*)^T(g(y^*) - B^T\lambda^*) \ge 0, \qquad \forall (x, y, \lambda) \in \Omega. \\ (\lambda - \lambda^*)^T(Ax^* + By^* - b) \ge 0, \end{cases} \tag{2.2}$$

are satisfied. Alternatively, (2.2) can be regarded as the first-order optimality condition of (1.2). Note that the variational reformulation (2.2) can be rewritten into a compact form. In fact, for $w = (x, y, \lambda) \in \Omega$, $f(x) \in \partial F(x)$ and $g(y) \in \partial G(y)$, if we denote

$$F(w) := \begin{pmatrix} f(x) - A^T\lambda \\ g(y) - B^T\lambda \\ Ax + By - b \end{pmatrix}, \tag{2.3}$$

then (2.2) can be rewritten as finding $w^* \in \Omega$, $f(x^*) \in \partial F(x^*)$ and $g(y^*) \in \partial G(y^*)$ such that

$$(w - w^*)^T F(w^*) \geq 0, \ \forall w \in \Omega. \tag{2.4}$$

As we shall show, with the variational reformulation (2.2) or (2.4), it is convenient for our theoretical analysis and it is easier to expose our motivation of developing the relaxed customized PPA. For many concrete applications of the model (1.2) including those to be tested in this paper, the decomposed subproblems in the optimization form are often easy enough to have closed-form solutions, and it is not necessary to choose any specific subgradient $f(x)$ or $g(y)$ in the implementation. Therefore, the variational reformulation (2.2) or (2.4) only serves as a theoretical tool in the coming analysis. Furthermore, we denote by Ω^* the set of such w^* that satisfies (2.2). Then, under the aforementioned nonempty assumption on the solution set of (1.2), obviously Ω^* is also nonempty.

Then, we review the application of the relaxed PPA in [13] to solving the variational reformulation (2.4). Let $w^k \in \Omega$ and $P \in \mathbb{R}^{(n+m)\times(n+m)}$ be a symmetric positive definite matrix, then the original PPA in [21, 28] generates the new iterate w^{k+1} via solving the regularized variational problem: finding $w^{k+1} \in \Omega$, $f(x^{k+1}) \in \partial F(x^{k+1})$ and $g(y^{k+1}) \in \partial G(y^{k+1})$ such that

$$(w - w^{k+1})^T \big(F(w^{k+1}) + P(w^{k+1} - w^k)\big) \geq 0, \ \forall w \in \Omega. \tag{2.5}$$

The relaxed PPA in [13], however, combines the PPA step (2.5) with a relaxation step. More precisely, let the solution of (2.5) be denoted by $\tilde{w}^k$, then the relaxed PPA in [13] yields the new iterate via

$$w^{k+1} = w^k - \gamma(w^k - \tilde{w}^k), \tag{2.6}$$

where $\gamma \in (0, 2)$ is the relaxation factor. In particular, γ is called an under-relaxation (when $\gamma \in (0, 1)$) or over-relaxation factor (when $\gamma \in (1, 2)$); and the relaxed PPA (2.6) reduces to the original PPA (2.5) when $\gamma = 1$.

Finally, we recall the ADM which was proposed originally in [12] and now is widely regarded as a benchmark solver for the model (1.2). More specifically, the iterative scheme of ADM for solving (1.2) is as follows

$$\begin{cases} x^{k+1} = \arg\min\{F(x) - (\lambda^k)^T(Ax + By^k - b) + \frac{\beta}{2}\|Ax + By^k - b\|^2 \mid x \in \mathcal{X}\}, \\ y^{k+1} = \arg\min\{G(y) - (\lambda^k)^T(Ax^{k+1} + By - b) + \frac{\beta}{2}\|Ax^{k+1} + By - b\|^2 \mid y \in \mathcal{Y}\}, \\ \lambda^{k+1} = \lambda^k - \beta(Ax^{k+1} + By^{k+1} - b), \end{cases} \tag{2.7}$$

where $\lambda \in \mathbb{R}^m$ is the Lagrange multiplier and $\beta > 0$ is the penalty parameter for the violation of the linear constraint. Obviously, if $(x^{k+1}, y^{k+1}, \lambda^{k+1})$ is generated by the ADM scheme (2.7), then there exists $f(x^{k+1}) \in \partial F(x^{k+1})$ and $g(y^{k+1}) \in \partial G(y^{k+1})$ such that

$$\begin{cases} (x - x^{k+1})^T(f(x^{k+1}) - A^T(\lambda^k - \beta(Ax^{k+1} + By^k - b)) \geq 0, \ \forall x \in \mathcal{X}, \\ (y - y^{k+1})^T(g(y^{k+1}) - B^T(\lambda^k - \beta(Ax^{k+1} + By^{k+1} - b)) \geq 0, \ \forall y \in \mathcal{Y}, \\ \lambda^{k+1} = \lambda^k - \beta(Ax^{k+1} + By^{k+1} - b). \end{cases} \tag{2.8}$$

The ADM (2.7) is a splitting version of the ALM where the ALM subproblem is decomposed into two subproblems in the Gauss-Seidel fashion at each iteration, and thus the variables x and y can be solved separably in the alternating order. Since the functions $F(x)$ and $G(y)$ often have specific properties for a particular application of (1.2), the decomposition treatment of ADM makes it possible to exploit these particular properties separately. In fact, the decomposed subproblems in (2.7) are often simple enough to have closed-form solutions or can be easily solved up to a high precision, and this feature contributes much to the recent burst of ADM's novel applications in various areas, see e.g. [6, 10, 16, 24, 30, 31] and references cited therein.

3 A relaxed customized proximal point algorithm

In this section, we propose a relaxed customized PPA for solving (1.2), preceded with an elaboration on the motivation.

Our idea comes from the fact that the PPA subproblem (2.5) is only of conceptual or theoretical interest, as it could be as difficult as the original variational problem (2.4) from the algorithmic implementation point of view. But, the specific choice of the proximal regularization matrix P enjoys favorable freedom and it enables us to choose a judicious one which is customized to the specific structure in (2.2). Here, we would emphasize that we do not require the positive definiteness of the proximal regularization matrix P. Instead, a positive semi-definite matrix is enough for our algorithmic design (see Remark 4.2). With only a positive semi-definite regularization matrix, it sounds not rigorous to still call (2.5) PPA. But, by this slight abuse of name, we only try to make our notation easier. In a word, with a customized choice of the proximal regularization matrix P, we expect that the PPA subproblem (2.5) is in such a decomposable form that the variables (x, y, λ) can be solved individually by exploiting the properties of $F(x)$ and $G(y)$ separately.

Because of the efficiency of ADM (2.7), our strategy of specifying the customized choice of P in (2.5) is to make the regularized variational problem (2.5) adherent to ADM to some extent. On the other hand, revisiting the iterative scheme (2.7), it is easy to observe that the variable x plays only an intermediate role and it is not involved in the execution of ADM, e.g. see the elaboration in [6]. Therefore, the input for executing the iteration of (2.7) is only the sequence $\{y^k, \lambda^k\}$. These facts inspire us to generate x^{k+1} exactly as ADM and to focus only on the proximal regularization on the variables (y, λ) in (2.5), i.e., the first n_1 rows of P are all zero. More concretely, x^{k+1} is generated via solving

$$x^{k+1} = \arg\min\{F(x) - (\lambda^k)^T(Ax + By^k - b) + \frac{\beta}{2}\|Ax + By^k - b\|^2 \mid x \in \mathcal{X}\}.$$

Accordingly, the above formula means there exists $f(x^{k+1}) \in \partial F(x^{k+1})$ such that

$$(x - x^{k+1})^T(f(x^{k+1}) - A^T(\lambda^k - \beta(Ax^{k+1} + By^k - b)) \geq 0,\ \forall x \in \mathcal{X}. \tag{3.1}$$

Moreover, recall that the x-related portion in the variational reformulation (2.2). After the step (3.1), if we update the Lagrange multiplier via

$$\lambda^{k+1} = \lambda^k - \beta(Ax^{k+1} + By^k - b), \tag{3.2}$$

then (3.1) can be written as

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

which implies that the x-related portion in the first-order optimality condition (2.2) is satisfied. This suggests us to update the Lagrange multiplier via (3.2) right after the computation of x^{k+1} by (3.1). By casting the scheme (3.2) into the form (2.5), it is then clear how to choose the λ-corresponding part of the regularization matrix P. In fact, (3.2) can be rewritten as

$$(\lambda - \lambda^{k+1})^T((Ax^{k+1} + By^{k+1} - b) - B(y^{k+1} - y^k) + \frac{1}{\beta}(\lambda^{k+1} - \lambda^k)) \geq 0,\ \forall \lambda \in \mathbb{R}^m.$$

Therefore, the scheme (3.2) essentially implies that the last m rows of the regularization matrix P in (2.5) is taken as $\left(-B, \frac{1}{\beta}I_m\right)$, where I_m denotes the $m \times m$ identity matrix.

Now, the rest is to specify the middle n_2 rows for the regularization matrix P in (2.5), i.e., the regularization on the variable y. In fact, with the specified regularization on x and λ, it is natural to specify the middle n_2 rows of G as $\left(\beta B^T B, -B^T\right)$ for the sake of ensuring the positive semi-definiteness of P. Then, with this choice, it follows from (2.5) that y^{k+1} should be generated by the following problem

$$(y - y^{k+1})^T (g(y^{k+1}) - B^T \lambda^{k+1} + \beta B^T B(y^{k+1} - y^k) - B^T(\lambda^{k+1} - \lambda^k)) \geq 0, \ \forall y \in \mathcal{Y}.$$

That is, y^{k+1} is generated via solving the following problem

$$y^{k+1} = \arg\min\{G(y) - (\lambda^{k+1})^T(Ax^{k+1} + By - b) + \frac{\beta}{2}\|Ax^{k+1} + By - b\|^2 \mid y \in \mathcal{Y}\}. \tag{3.3}$$

In a summary, in (2.5) we propose to choose a customized proximal regularization matrix as

$$P = \begin{pmatrix} 0 & 0 \\ \beta B^T B & -B^T \\ -B & \frac{1}{\beta} I_m \end{pmatrix} \tag{3.4}$$

and the resulting customized PPA for (1.2) has the following algorithmic framework.

Algorithm 1: A customized proximal point algorithm for (1.2)

Let $\beta > 0$. With the given iterate w^k, the new iterate w^{k+1} is generated as follows.

$$\left\{\begin{array}{l} x^{k+1} = \arg\min\{F(x) - (\lambda^k)^T(Ax + By^k - b) + \frac{\beta}{2}\|Ax + By^k - b\|^2 \mid x \in \mathcal{X}\}; \\ \lambda^{k+1} = \lambda^k - \beta(Ax^{k+1} + By^k - b); \\ y^{k+1} = \arg\min\{G(y) - (\lambda^{k+1})^T(Ax^{k+1} + By - b) + \frac{\beta}{2}\|Ax^{k+1} + By - b\|^2 \mid y \in \mathcal{Y}\}. \end{array}\right. \tag{3.5}$$

Comparing with the ADM scheme (2.7), the customized PPA (3.5) yields the new iterate in the order $x \to \lambda \to y$ with the difference in the order of λ and y. Despite this difference, the customized PPA (3.5) and the ADM (2.7) are equally effective to exploit the separable structure of (1.2) and equally easy to implement. In fact, the resulting subproblems of these two methods are of the same degree of decomposition and they are of the same difficulty. In Section 5, we shall verify by numerical evidence that these two methods are also equally competitive in numerical senses.

Before we present the relaxed customized PPA, we would reiterate that the variable x plays only an intermediate role and it is not involved in the execution of (2.7) or (3.5). Accordingly, the following notations regarding the variables y and λ will simplify our analysis

$$v = (y, \lambda); \ \ \mathcal{V} = \mathcal{Y} \times \mathbb{R}^m;$$

$$v^k = (y^k, \lambda^k); \quad \tilde{v}^k = (\tilde{y}^k, \lambda^k), \ \forall k \in \mathcal{N};$$

$$v^* = (y^*, \lambda^*); \ \ \mathcal{V}^* = \{(y^*, \lambda^*) \,|\, (x^*, y^*, \lambda^*) \in \Omega^*\}.$$

As we have mentioned, inspired by [13], we would combine the customized PPA (3.5) with the relaxation step (2.6). Therefore, a relaxed customized PPA is proposed to solve (1.2). Instead of $(x^{k+1}, y^{k+1}, \lambda^{k+1})$ in (3.5), the output of the customized PPA is labeled as $(\tilde{x}^k, \tilde{y}^k, \tilde{\lambda}^k)$ below. Since the variable x is an intermediate variable and it is not required by the iteration, the relaxation step (2.6) is only implemented for the variables y and λ. That is why we only relax v^k, instead of w^k, in the relaxation step (3.7) of the proposing relaxed customized PPA.

Algorithm 2: A relaxed customized proximal point algorithm for (1.2)

Let $\beta > 0$ and $\gamma \in (0, 2)$. With the given iterate w^k, the new iterate w^{k+1} is generated as follows.

Step 1. Customized PPA Step. Obtain $\tilde{w}^k = (\tilde{x}^k, \tilde{y}^k, \tilde{\lambda}^k)$ via

$$\begin{cases} \tilde{x}^k = \arg\min\{F(x) - (\lambda^k)^T(Ax + By^k - b) + \frac{\beta}{2}\|Ax + By^k - b\|^2 \mid x \in \mathcal{X}\}; \\ \tilde{\lambda}^k = \lambda^k - \beta(A\tilde{x}^k + By^k - b); \\ \tilde{y}^k = \arg\min\{G(y) - (\tilde{\lambda}^k)^T(A\tilde{x}^k + By - b) + \frac{\beta}{2}\|A\tilde{x}^k + By - b\|^2 \mid y \in \mathcal{Y}\}; \end{cases} \tag{3.6}$$

Step 2. Relaxation Step.

$$v^{k+1} = v^k - \gamma(v^k - \tilde{v}^k). \tag{3.7}$$

Remark 3.1. For the relaxation step (3.7), it is only a simple linear combination of the vectors v^k and $\tilde{v}^k$. Therefore, the additional computation resulted by this step is almost null, and thus the x- and y-subproblems dominate the computation of each iteration of the proposed method. The simplicity of the relaxation factor outperforms some descent-like steps combined with ADM which requires demanding computation for finding appropriate step sizes (e.g. [35]). In addition, for the relaxation factor, as we shall show later both theoretically in the analysis of the convergence rate (see Theorem 5.3) and numerically in Section 6, aggressive values such as $\gamma \in (1.5, 2)$ are often preferred. That is, the over-relaxation with $\gamma \in (1, 2)$ is better than the under-relaxation with $\gamma \in (0, 1)$, supporting the conjectured assertion posed in [3].

Remark 3.2. Obviously, the customized PPA (3.5) is a special case of the proposed Algorithm 2 with $\gamma = 1$ in (3.7). Therefore, the coming convergence analysis is only conducted for Algorithm 2.

4 Convergence

In this section, we prove the global convergence of the proposed Algorithm 2. The proof follows the analytic framework of contraction type methods (see [4] for the definition).

We start the proof with a lemma which is useful for the analysis of coming theorems.

Lemma 4.1. *Let the sequences $\{v^k\}$ and $\{\tilde{v}^k\}$ be generated by the proposed Algorithm 2. Then, we have*

$$(v^k - v^*)^T M(v^k - \tilde{v}^k) \geq (v^k - \tilde{v}^k)^T M(v^k - \tilde{v}^k), \ \ \forall v^* \in \mathcal{V}^*, \tag{4.1}$$

where

$$M = \begin{pmatrix} \beta B^T B & -B^T \\ -B & \frac{1}{\beta} I_m \end{pmatrix}. \tag{4.2}$$

Proof. As we have elaborated in Section 3, the customized PPA step (3.6) means there exist $f(\tilde{x}^k) \in \partial F(\tilde{x}^k)$ and $g(\tilde{y}^k) \in \partial G(\tilde{y}^k)$ such that

$$\begin{pmatrix} x - \tilde{x}^k \\ y - \tilde{y}^k \\ \lambda - \tilde{\lambda}^k \end{pmatrix}^T \left\{ \begin{pmatrix} f(\tilde{x}^k) - A^T\tilde{\lambda}^k \\ g(\tilde{y}^k) - B^T\tilde{\lambda}^k \\ A\tilde{x}^k + B\tilde{y}^k - b \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ \beta B^T B & -B^T \\ -B & \frac{1}{\beta} I_m \end{pmatrix} \begin{pmatrix} \tilde{y}^k - y^k \\ \tilde{\lambda}^k - \lambda^k \end{pmatrix} \right\} \geq 0, \ \forall (x, y, \lambda) \in \Omega, \tag{4.3}$$

or more compactly,

$$(w - \tilde{w}^k)^T(F(\tilde{w}^k) + M(\tilde{v}^k - v^k)) \geq 0, \ \forall w \in \Omega. \tag{4.4}$$

Recall that

$$F(\tilde{w}^k) := \begin{pmatrix} f(\tilde{x}^k) - A^T\tilde{\lambda}^k \\ g(\tilde{y}^k) - B^T\tilde{\lambda}^k \\ A\tilde{x}^k + B\tilde{y}^k - b \end{pmatrix}.$$

Choosing $w = (x, y, \lambda)$ as $w^* = (x^*, y^*, \lambda^*) \in \Omega^*$ in (4.3), then (4.4) can be rewritten as

$$(\tilde{v}^k - v^*)^T M(v^k - \tilde{v}^k) \geq (\tilde{w}^k - w^*)^T F(\tilde{w}^k).$$

Recall that $w^* \in \Omega^*$. Since the subdifferential operators $\partial F(x)$ and $\partial G(y)$ are monotone, we have that

$$(\tilde{w}^k - w^*)^T F(\tilde{w}^k) \geq 0.$$

Therefore, the above two inequalities imply that

$$(\tilde{v}^k - v^*)^T M(v^k - \tilde{v}^k) \geq 0,$$

from which the assertion (4.1) is immediately derived. □

Remark 4.2. Note that the matrix M defined in (4.2) is only positive semi-definite without further assumption on the matrix B. For notational convenience, by a slight abuse, throughout we still use the notation

$$\|v^k - \tilde{v}^k\|_M := \sqrt{(v^k - \tilde{v}^k)^T M(v^k - \tilde{v}^k)}.$$

Then, with this notation, the assertion (4.1) can be rewritten as

$$(v^k - v^*)^T M(v^k - \tilde{v}^k) \geq \|v^k - \tilde{v}^k\|_M^2 \geq 0, \ \ \forall v^* \in \mathcal{V}^*.$$

Remark 4.3. The inequality (4.1) implies that the direction $-(v^k - \tilde{v}^k)$ is beneficial for reducing the function value $\frac{1}{2}\|v - v^*\|_M^2$ at $v = v^k$ for $v^* \in \mathcal{V}^*$. Therefore, it justifies, at least intuitively, that the relaxation step (3.7) is effective for reducing the proximity toward the set $\mathcal{V}^*$.

Now we are ready to show a useful property of the sequence $\{v^k\}$ in the following lemma.

Lemma 4.4. *Let the sequences $\{v^k\}$ and $\{\tilde{v}^k\}$ be generated by the proposed Algorithm 2 and the matrix M be defined in (4.2). Then, we have*

$$\|v^{k+1} - v^*\|_M^2 \leq \|v^k - v^*\|_M^2 - \gamma(2-\gamma)\|v^k - \tilde{v}^k\|_M^2, \ \ \forall v^* \in \mathcal{V}^*. \tag{4.5}$$

Proof. By elementary manipulation, we obtain

$$\begin{aligned} \|v^{k+1} - v^*\|_M^2 &= \|(v^k - v^*) - \gamma(v^k - \tilde{v}^k)\|_M^2 \\ &= \|v^k - v^*\|_M^2 - 2\gamma(v^k - v^*)^T M(v^k - \tilde{v}^k) + \gamma^2\|v^k - \tilde{v}^k\|_M^2 \\ &\leq \|v^k - v^*\|_M^2 - 2\gamma\|v^k - \tilde{v}^k\|_M^2 + \gamma^2\|v^k - \tilde{v}^k\|_M^2 \\ &= \|v^k - v^*\|_M^2 - \gamma(2-\gamma)\|v^k - \tilde{v}^k\|_M^2, \end{aligned}$$

where the inequality follows from (4.1). □

With the above fundamental theorem, the convergence result of the proposed Algorithm 2 follows easily.

Theorem 4.5. *Let the sequences $\{v^k\}$ and $\{\tilde{v}^k\}$ be generated by the proposed Algorithm 2 and the matrix M be defined in (4.2). Then, we have*

(i) $\lim_{k\to\infty}\{\|v^k - \tilde{v}^k\|_M\} = 0$;

(ii) Any accumulation point of $\{\tilde{w}^k\}$ is a solution point of the problem (1.2).

Proof. The first assertion is an immediate conclusion of (4.5) with the restriction $\gamma \in (0, 2)$. Recall that the matrix M is symmetric and positive semi-definite. Thus, from assertion (i), we have that $M(v^k - \tilde{v}^k) = 0$. By substituting $M(v^k - \tilde{v}^k) = 0$ into (4.3) or its compact form (2.5), we obtain that

$$(w - \tilde{w}^k)^T F(\tilde{w}^k) \geq 0, \ \forall w \in \Omega,$$

which means $\tilde{w}^k = (\tilde{x}^k, \tilde{y}^k, \tilde{\lambda}^k)$ satisfies the first-order optimal condition of problem (1.2). Hence, the assertion (ii) follows. □

Remark 4.6. Recall that the model (1.2) may have more than one solution. Thus, the accumulation point of the sequence $\{\tilde{w}^k\}$ may not be unique. If the subproblem (3.3) is regularized proximally by the term $\frac{\delta}{2}\|y - y^k\|^2$ where the proximal parameter δ can be arbitrarily small, then accordingly the matrix M in (4.2) is altered to

$$\begin{pmatrix} \beta B^T B + \delta I_{n_2} & -B^T \\ -B & \frac{1}{\beta} I_m \end{pmatrix},$$

which is positive definite. Correspondingly, it is easy to prove that the generated sequence $\{\tilde{w}^k\}$ has only one accumulation point, i.e., it is convergent.

5 Convergence rate

In this section, we derive the $O(1/t)$ convergence rate for the proposed Algorithm 2. The $O(1/t)$ convergence rate of the original PPA has been shown in the literature, e.g., [22, 32], but our result is for the relaxed customized PPA where the relaxation factor $\gamma \in (0, 2)$. Moreover, our proof is based on a variational characterization of (1.2), and it is much simpler than the existing ones. As we have mentioned, the simple proof of the $O(1/t)$ convergence rate also gives a hint why the relaxation factor is preferred in $(1, 2)$ for the purpose of accelerating the customized PPA. Thus, we provide a theoretical support on the conjectured assertion in [3] of choosing an over-relaxation factor in $(1, 2)$ empirically.

As a preparation for the proof of the $O(1/t)$ convergence rate, we first recall a variational characterization of the solution set of (1.2) in [11] (Theorem 2.3.5 therein). For completeness, we still include the proof.

Lemma 5.1. *For $w = (x, y, \lambda)$, $f(x) \in \partial F(x)$ and $g(y) \in \partial G(y)$, let $F(w)$ be given by (2.3). Then, the solution set Ω^* of (2.4) is convex and it can be characterized as*

$$\Omega^* = \{w^* \in \Omega \mid (w - w^*)^T F(w) \geq 0, \ \forall w \in \Omega\}. \tag{5.1}$$

Proof. For $w^* \in \Omega^*$, we have

$$(w - w^*)^T F(w^*) \geq 0, \ \forall w \in \Omega.$$

Recall the monotonicity of the subdifferential operator of a convex function, we thus obtain

$$(w - w^*)^T F(w) \geq 0, \ \forall w \in \Omega.$$

Therefore, we show $\Omega^* \subseteq \{w^* \in \Omega \mid (w - w^*)^T F(w) \geq 0, \ \forall w \in \Omega\}$.

Conversely, let $w^* \in \{w^* \in \Omega \mid (w - w^*)^T F(w) \geq 0, \ \forall w \in \Omega\}$. For an arbitrary $w \in \Omega$, the vector

$$w' := \tau w^* + (1 - \tau) w$$

belongs to Ω for all $\tau \in (0,1)$. Thus we have

$$(w' - w^*)F(w') \geq 0.$$

By substitution, we derive that

$$(1-\tau)(w - w^*)^T F(\tau w^* + (1-\tau)w) \geq 0, \ \forall \tau \in (0,1).$$

Letting $\tau \to 1$ yields

$$(w - w^*)^T F(w^*) \geq 0, \ \forall w \in \Omega.$$

Hence $w^* \in \Omega^*$, and we obtain $\{w^* \in \Omega \mid (w - w^*)^T F(w) \geq 0, \ \forall w \in \Omega\} \subseteq \Omega^*$. The identity (5.1) is thus established.

Finally, based on (5.1), it is trivial to prove the convexity of Ω^* and we omit it. $\square$

To show the $O(1/t)$ convergence rate of the proposed Algorithm 2, we need to prove an inequality in the following lemma.

Lemma 5.2. *Let the sequences $\{v^k\}$, $\{\tilde{v}^k\}$ and $\{\tilde{w}^k\}$ be generated by the proposed Algorithm 2 and the matrix M be defined in (4.2). Then, we have*

$$(w - \tilde{w}^k)^T F(\tilde{w}^k) + \frac{1}{2\gamma}(\|v - v^k\|_M^2 - \|v - v^{k+1}\|_M^2) \geq (1 - \frac{\gamma}{2})\|v^k - \tilde{v}^k\|_M^2, \ \forall w \in \Omega.$$

Proof. It follows from (4.3) that

$$(w - \tilde{w}^k)^T F(\tilde{w}^k) \geq (v - \tilde{v}^k)^T M(v^k - \tilde{v}^k), \ \forall w \in \Omega.$$

Thus, it suffices to show that

$$(v - \tilde{v}^k)^T M(v^k - \tilde{v}^k) + \frac{1}{2\gamma}(\|v - v^k\|_M^2 - \|v - v^{k+1}\|_M^2) \geq (1 - \frac{\gamma}{2})\|v^k - \tilde{v}^k\|_M^2. \tag{5.2}$$

By using the formula $a^T M b = \frac{1}{2}(\|a\|_M^2 + \|b\|_M^2 - \|a - b\|_M^2)$, we derive that

$$(v - v^{k+1})^T M(v^k - v^{k+1}) = \frac{1}{2}\|v - v^{k+1}\|_M^2 + \frac{1}{2}\|v^k - v^{k+1}\|_M^2 - \frac{1}{2}\|v - v^k\|_M^2.$$

Moreover, since the relaxation step (3.7) can be rewritten as $(v^k - v^{k+1}) = \gamma(v^k - \tilde{v}^k)$, we have

$$(v - v^{k+1})^T M(v^k - \tilde{v}^k) = \frac{1}{\gamma}(v - v^{k+1})^T M(v^k - v^{k+1}).$$

Combining the last two equations, we obtain

$$(v - v^{k+1})^T M(v^k - \tilde{v}^k) = \frac{1}{2\gamma}(\|v - v^{k+1}\|_M^2 - \|v - v^k\|_M^2) + \frac{1}{2\gamma}\|v^k - v^{k+1}\|_M^2. \tag{5.3}$$

On the other hand, we have

$$(v^{k+1} - \tilde{v}^k)^T M(v^k - \tilde{v}^k) = [(v^k - \tilde{v}^k) - \gamma(v^k - \tilde{v}^k)]^T M(v^k - \tilde{v}^k) = (1-\gamma)\|v^k - \tilde{v}^k\|_M^2, \tag{5.4}$$

where (3.7) is used in the first equality. By adding (5.3) and (5.4), and again using the fact that $(v^k - v^{k+1}) = \gamma(v^k - \tilde{v}^k)$, we obtain that

$$\begin{aligned}
&(v - \tilde{v}^k)^T M(v^k - \tilde{v}^k) \\
&= \frac{1}{2\gamma}(\|v - v^{k+1}\|_M^2 - \|v - v^k\|_M^2) + \frac{1}{2\gamma}\|v^k - v^{k+1}\|_M^2 + (1-\gamma)\|v^k - \tilde{v}^k\|_M^2 \\
&= \frac{1}{2\gamma}(\|v - v^{k+1}\|_M^2 - \|v - v^k\|_M^2) + (1 - \frac{\gamma}{2})\|v^k - \tilde{v}^k\|_M^2,
\end{aligned}$$

which is equivalent to (5.2). Hence, the lemma is proved. $\square$

Now, we are ready to prove the $O(1/t)$ convergence rate in an ergodic sense for the proposed Algorithm 2.

Theorem 5.3. *Let the sequences $\{\tilde{w}^k\}$ be generated by the proposed Algorithm 2. For an integer $t > 0$, let*

$$\bar{w}_t := \frac{1}{t+1}\sum_{k=0}^{t} \tilde{w}^k, \tag{5.5}$$

then $\bar{w}_t \in \Omega$ and

$$(\bar{w}_t - w)^T F(w) \leq \frac{1}{2\gamma(t+1)}\|v - v^0\|_M^2,\ \forall w \in \Omega. \tag{5.6}$$

Proof. First, for an integer $t > 0$, we have $\tilde{w}^k = (\tilde{x}^k, \tilde{y}^k, \tilde{\lambda}^k) \in \Omega$ for $k = 0, 1, \cdots, t$. Since $\frac{1}{t+1}\sum_{k=0}^{t} \tilde{w}^k$ can be viewed as a convex combination of $\tilde{w}^k$'s, we obtain $\bar{w}_t = \frac{1}{t+1}\sum_{k=0}^{l} \tilde{w}^k \in \Omega$.

Second, since $\gamma \in (0, 2)$, it follows from Lemma 5.2 that

$$(w - \tilde{w}^k)^T F(\tilde{w}^k) + \frac{1}{2\gamma}(\|v - v^k\|_M^2 - \|v - v^{k+1}\|_M^2) \geq 0,\ \forall w \in \Omega.$$

By combining the monotonicity of $F(\cdot)$ with the last inequality, we obtain

$$(w - \tilde{w}^k)^T F(w) + \frac{1}{2\gamma}(\|v - v^k\|_M^2 - \|v - v^{k+1}\|_M^2) \geq 0,\ \forall w \in \Omega.$$

Summing the above inequality over $k = 0, 1, \ldots, t$, we derive that

$$\left((t+1)w - \sum_{k=0}^{t} \tilde{w}^k\right)^T F(w) + \frac{1}{2\gamma}(\|v - v^0\|_M^2 - \|v - v^{t+1}\|_M^2) \geq 0,\ \forall w \in \Omega.$$

By dropping the minus term, we have

$$\left((t+1)w - \sum_{k=0}^{t} \tilde{w}^k\right)^T F(w) + \frac{1}{2\gamma}\|v - v^0\|_M^2 \geq 0,\ \forall w \in \Omega,$$

which is equivalent to

$$\left(\frac{1}{t+1}\sum_{k=0}^{t} \tilde{w}^k - w\right)^T F(w) \leq \frac{1}{2\gamma(t+1)}\|v - v^0\|_M^2,\ \forall w \in \Omega.$$

The proof is completed. □

According to Theorem 5.3, for any given compact set $\mathcal{D} \subset \Omega$, let $d := \sup\{\|v - v^0\|_M^2 \mid w \in \mathcal{D}\}$. Then, after t iterations of the proposed Algorithm 2, the point $\bar{w}_t$ defined in (5.5) satisfies

$$\sup_{w \in \mathcal{D}}\{(\bar{w}_t - w)^T F(w)\} \leq \frac{d}{2(t+1)},$$

which means that $\bar{w}_t$ is an approximate solution of (2.2) with the accuracy $O(1/t)$. That is, the convergence rate $O(1/t)$ of the proposed Algorithm 2 is established in an ergodic sense.

Last, we would remark on the choice of the relaxation factor γ. According to (5.6), it is obvious that larger values of γ is more beneficial for accelerating the convergence of the proposed Algorithm 2. Considering the requirement $\gamma \in (0, 2)$ (see (4.5)), it is preferred to choose $\gamma \to 2$ empirically. This is a theoretical support on the suggested choice posed in [3]. In Section 6, we will show by numerical results that an aggressive choice $\gamma \to 2$ does accelerate the convergence of the proposed Algorithm 2.

6 Numerical results

In this section, we report some preliminary numerical results to verify the facts: 1) the customized PPA (3.5) is competitive to the ADM (2.7); 2) the relaxation step (3.7) with an over-relaxation factor can accelerate the customized PPA (3.5) easily, and thus the proposed Algorithm 2 outperforms some existing methods including the ADM (2.7) and the method in [35]. All the code were written by MATLAB 7.9 and were performed on a Dell computer equipped with Windows XP, 2.66 GHz IntelCore 2 Duo CPU and 2GB of memory.

6.1 The least squares semi-definite programming problem

In this subsection, we apply the proposed Algorithm 2 to solve the least squares semi-definite programming (LSSDP) problem, and compare it with the ADM (2.7) and the ADM-based descent method in [35] (denoted by "YY" according to the acronym of authors) which aims at accelerating the ADM (2.7) via a descent step. By this example, we do not go into details for the acceleration performance contributed by the relaxation factor in (3.7), but only focus on the illustration that Algorithm 2 with an appropriate relaxation factor can be as efficient as, or even better than, the ADM (2.7) and the YY method.

Mathematically, the LSSDP problem is

$$\min\{\frac{1}{2}\|X-C\|_F^2 \mid X \in S_+^n \cap S_B\}, \tag{6.1}$$

where C is a given $n \times n$ symmetric matrix; S_+^n is the cone of symmetric positive semi-definite matrices; $\|\cdot\|_F$ is the standard Frobenius norm and S_B represents the component-wise constraints on X, i.e.,

$$S_B = \{H \in \mathbb{R}^{n\times n} \mid H_L \le H \le H_U\},$$

where H_L and H_U are two given $n \times n$ symmetric matrices. We refer to, e.g. [7, 5, 19, 20] and references cited therein, for the applications of LSSDP arising in various areas such as finance, statistics, and machine learning.

As in [16], by introducing an auxiliary variable Y, (6.1) can be reformulated as the following form for which the ADM (2.7) is readily applicable

$$\begin{aligned} \min\ & \frac{1}{2}\|X-C\|_F^2 + \frac{1}{2}\|Y-C\|_F^2 \\ s.t.\ & X - Y = 0, \\ & X \in S_+^n,\ Y \in S_B. \end{aligned} \tag{6.2}$$

Note that, (6.2) is a special case of the extension of (1.2) with matrix variables. Obviously, all the previous analysis can be straightforwardly extended to the extension of (1.2) with matrix variables, and we thus omit the detail. For this subsection, the variables (x, y, λ) in the proposed Algorithm 2 are replaced by (X, Y, Λ) accordingly for denoting the matrix variables.

More specifically, (6.2) is a special case of (1.2) with $F(X) = \frac{1}{2}\|X-C\|_F^2$ and $G(Y) = \frac{1}{2}\|Y-C\|_F^2$; $A = \mathcal{I}$ and $B = -\mathcal{I}$ where $\mathcal{I}$ stands for the identity mapping; $\mathcal{X} = S_+^n$ and $\mathcal{Y} = S_B$. Accordingly, below we elaborate on how to derive the closed-form solutions for the subproblems resulted by the customized PPA (3.6).

- With the given Y^k and Λ^k, $\tilde{X}^k$ is calculated via (3.6), i.e.,

$$\tilde{X}^k = \arg\min\{\frac{1}{2}\|X-C\|_F^2 - \mathrm{Tr}(\Lambda^k X) + \frac{\beta}{2}\|X-Y^k\|_F^2 \mid X \in S_+^n\},$$

where "Tr" denotes the standard trace of a matrix. As a consequence, $\tilde{X}^k$ has the following closed form

$$\tilde{X}^k = P_{S_+^n}\{\frac{1}{1+\beta}(\beta Y^k + Z^k + C)\},$$

where $P_{S_+^n}\{\cdot\}$ denotes the projection onto the positive semidefinite cone under the Euclidean norm (which can be computed via the eigenvalue decomposition).

- Then, with the newly generated $\tilde{X}^k$, the Lagrange multiplier is updated via

$$\tilde{\Lambda}^k = \Lambda^k - \beta(\tilde{X}^k - Y^k).$$

- With the updated $\tilde{X}^k$ and $\tilde{\Lambda}^k$, the variable Y is updated via

$$\tilde{Y}^k = \arg\min\{\frac{1}{2}\|Y - C\|_F^2 + \mathrm{Tr}(\tilde{\Lambda}^k Y) + \frac{\beta}{2}\|\tilde{X}^k - Y\|_F^2 \mid Y \in S_B\},$$

and its closed-form solution is given by

$$\tilde{Y}^k = P_{S_B}\{\frac{1}{1+\beta}(\beta\tilde{X}^k - \tilde{\Lambda}^k + C)\},$$

where P_{S_B} denotes the projection onto S_B under the Euclidean norm.

For the numerical experiments, we test some synthetic data for the model (6.1) in different dimensionality n. More specifically, the off-diagonal entries of C are randomly generated in the interval $(-1, 1)$, while its diagonal entries are randomly chosen in $(0, 2)$. For the bounding matrices H_L and H_U, their off-diagonal entries are randomly generated in the interval $(-1, 1)$ and their diagonal entries are all 1's.

We compare the proposed Algorithm 2 with the original ADM (2.7) and the YY method in [35]. To implementation these algorithms, the penalty parameter β is taken as 10 throughout; all the iterations start with the initial iterate $Y^0 = \mathcal{I}$ and $\Lambda^0 = \mathbf{0}$; and the stopping criterion is

$$\mathrm{Tol} := \max_{1\le i,j\le n} |Y_{i,j} - \tilde{Y}_{i,j}| + \max_{1\le i,j\le n} |\Lambda_{i,j} - \tilde{\Lambda}_{i,j}| \le 10^{-5}.$$

For the YY method, we take the relaxation factor as 1.8 in its descent step. In Table 1, for various scenarios of n, we report the number of iterations ("Iter.") and computing time in seconds ("CPU") for these mentioned methods.

The data in Table 1 shows the efficiency of all the tested methods for solving the LSSDP (6.1), even for large-scale cases. Our theoretical assertions are verified by this table. More concretely, 1) the customized PPA (3.5), i.e., the proposed Algorithm 2 with $\gamma = 1$, performs almost equally efficiently as the original ADM (2.7); and 2) the over-relaxation factor in (3.7) is effective to accelerate the customized PPA easily. In fact, even compared to the YY method with an intensively chosen step size, the proposed Algorithm 2 with $\gamma = 1.5$ performs better.

Note that the YY method also involves a relaxation factor in its descent step. In general, an appropriate choice for this factor is problem-dependent and it requires to tune in the numerical implementation. We have tested a number of values for this factor and we found that the proposed Algorithm 2 with $\gamma = 1.5$ is even not worse than the empirically best performance of the YY method with an optimally tuned relaxation factor. Recall that the proposed Algorithm 2 is free from any demanding additional computation for seeking an appropriate step sizes at the relaxation step, while the YY method might require considerate computation for this purpose. Therefore, the implementation of the proposed Algorithm 2 is easier than that of the YY method, and the numerical efficiency of the former is better than the latter.

Table 1: Numerical comparison of ADM, YY method and Algorithm 2 for (6.1).

	ADM (2.7)		Algo2, $\gamma = 1.0$		YY in [35]		Algo2, $\gamma = 1.5$	
n	Iter	CPU	Iter	CPU	Iter	CPU	Iter	CPU
25	48	0.02	46	0.02	37	0.02	30	0.01
50	47	0.06	47	0.06	43	0.06	36	0.05
100	49	0.22	51	0.23	38	0.18	35	0.16
200	46	0.84	51	0.95	32	0.62	35	0.66
300	48	2.28	52	2.60	33	1.75	34	1.70
400	51	4.84	54	5.23	38	4.01	35	3.38
500	52	8.47	55	8.99	41	7.30	35	5.73
600	53	13.45	56	14.46	44	12.25	37	9.55
700	54	20.17	57	21.53	46	18.83	37	13.96
800	55	29.12	57	30.88	47	28.64	37	20.05
1000	57	55.98	59	58.83	48	51.49	38	39.26
1200	58	97.88	61	100.73	49	88.47	40	65.93
1500	60	186.21	62	193.34	50	163.89	41	127.81
2000	64	458.90	65	468.33	50	374.56	42	303.51

6.2 Total variation image restoration problem

In this subsection, we apply the proposed Algorithm 2 to solve another interesting problem: the total variation image restoration problem arising in the discipline of image processing. Our theoretical assertions will be further verified by this example.

We first briefly review the background of the digital image restoration problem, and we refer to [14, 26] for more details. Let $x \in \mathbb{R}^n$ denote a two-dimensional image. Let $n = n_1 \cdot n_2$ be the total number of pixels, where n_1 and n_2 denote the numbers of pixels in the horizontal and vertical directions, respectively. The image restoration problem is to restore the original image $\bar{x}$ from its degraded image, denoted by x^0, and the mathematical model is

$$x^0 = K\bar{x} + \omega. \tag{6.3}$$

where $\omega \in \mathbb{R}^m$ is an additive noise corrupting the original image and $K : \mathbb{R}^n \to \mathbb{R}^m$ is a linear transform (e.g. a convolution by a blurring kernel). Since the model (6.3) is usually ill-posed, certain regularization techniques are required. One of the most popular regularization techniques is the total variation (TV) regularization proposed in the seminal work [29], mainly because of its capability of preserving the edges of images. Let $\partial_1 : \mathbb{R}^n \to \mathbb{R}^n$ and $\partial_2 : \mathbb{R}^n \to \mathbb{R}^n$ be the finite-difference operators in the horizontal and vertical directions, respectively; and let $\nabla := (\partial_1, \partial_2)$ denote the gradient operator. As [1, 2, 33], we consider the TV-l^2 model

$$\min |||\nabla x|||_1 + \frac{\mu}{2}\|Kx - x^0\|^2, \tag{6.4}$$

where $\mu > 0$ is a trade-off constant between the TV regularization and data-fidelity terms; and the l^1-norm $|||\cdot|||_1$ defined on $\mathbb{R}^n \times \mathbb{R}^n$ is given by

$$|||y|||_1 := \|(|y|)\|_1, \quad \forall y = (y_1, y_2) \in \mathbb{R}^n \times \mathbb{R}^n. \tag{6.5}$$

Here, $|y| := \sqrt{y_1^2 + y_2^2} \in \mathbb{R}^n$ is understood in the componentwise sense: $(|y|)_i := \sqrt{(y_1)_i^2 + (y_2)_i^2}$, see e.g. [26, Chapter 1].

By introducing the auxiliary variables $y \in \mathbb{R}^n \times \mathbb{R}^n$, we can reformulate (6.4) as

$$\begin{aligned} &\min \, |||y|||_1 + \frac{\mu}{2}\|Kx - x^0\|^2 \\ &\text{s.t. } \nabla x = y, \end{aligned} \tag{6.6}$$

which is a special case of the model (1.2) with $F(x) = \frac{\mu}{2}\|Kx - x^0\|^2$, $G(y) = |||y|||_1$, $b = 0$, $A = \nabla$ and $B = -I$. Therefore, the original ADM (2.7) is applicable and it is essentially the FTVd in [34].

Now, we illustrate that all the resulting subproblems (3.6) have closed-form solutions when the proposed Algorithm 2 is applied to solve (6.6). At each iteration, with the given y^k and λ^k, the customized PPA step (3.6) consists of the following subproblems.

- The variable $\tilde{x}^k$ is calculated via (see (3.6))

  $$\tilde{x}^k = \arg\min_x \left\{ \frac{\mu}{2}\|Kx - x^0\|^2 - (\lambda^k)^T(\nabla x - y^k) + \frac{\beta}{2}\|\nabla x - y^k\|^2 \right\}.$$

  The above minimization problem amounts to a least-squares problem whose normal equation is given by

  $$(\beta \nabla^T \nabla + \mu K^T K)\tilde{x}^k = \nabla^T(\beta y^k + \lambda^k) + \mu K^T x^0. \tag{6.7}$$

  In particular, when K is a spatially invariant convolution operator and the periodic boundary conditions are used, the operators K and ∇ can be diagonalized by the discrete Fourier transform (DFT)

  $$K = \mathcal{F}^{-1} D_K \mathcal{F} \quad \text{and} \quad \nabla = \mathcal{F}^{-1} D \mathcal{F},$$

  where D_K and D are diagonal matrices with positive diagonal entries and $\mathcal{F}$ is the DFT operator [14]. Thus, the variable $\tilde{u}^k$ in (6.7) can be easily obtained by a DFT and an inverse DFT, see e.g. [23].

- The variable $\tilde{\lambda}^k$ is updated by

  $$\tilde{\lambda}^k = \lambda^k - \beta(\nabla \tilde{x}^k - y^k).$$

- The variable $\tilde{y}^k$ is updated by

  $$\tilde{y}^k = \arg\min_y \left\{ |||y|||_1 - (\tilde{\lambda}^k)^T(\nabla \tilde{x}^k - y) + \frac{\beta}{2}\|\nabla \tilde{x}^k - y\|^2 \right\}.$$

  Note that the closed-form solution of $\tilde{y}^k$ is given by

  $$\tilde{y}^k = \text{shrink}_{\frac{1}{\beta}}\left(\nabla \tilde{x}^k - \frac{\tilde{\lambda}^k}{\beta} \right),$$

  where the multidimensional shrinkage operator is defined as $\text{shrink}_{\frac{1}{\beta}}(t) = t - \min\{\frac{1}{\beta}, \|t\|\}\frac{t}{\|t\|}$, see e.g. [33].

For the numerical experiments, we test the grayscale image chart.tif (256 × 256) and the color image house.png (256 × 256). We degrade both the images by blurring it with the out-of-focus kernel (radius as 7) and adding the zero-mean white Gaussian noise with the standard deviation 0.01 and 0.02, respectively. See Figure 1 for the original images and degraded images.

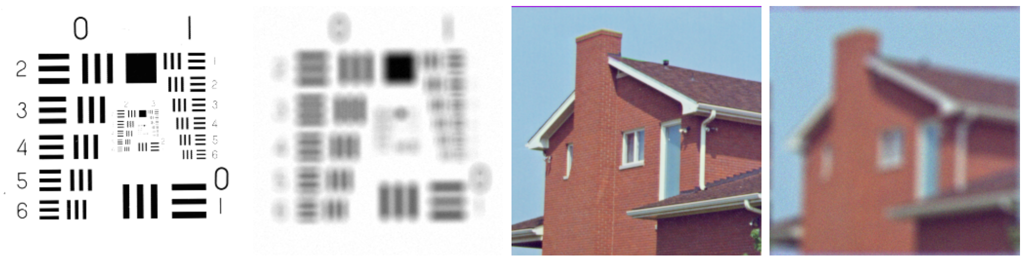
Figure 1: Original and degraded images. Left: chart.tif; Right: house.png

To restore these images via the model (6.4), we take $\mu = 10^3$ in (6.4). We apply both the original ADM (2.7) and the proposed Algorithm 2. For the parameters of the tested algorithms, we take $\beta = 30$ in both ADM and the proposed method. In the literature, the signal-to-noise ratio (SNR) in the unit of dB is usually used to measure the quality of restored images, and is defined by

$$\mathrm{SNR} = 20 \log_{10} \frac{\|x\|}{\|x - \bar{x}\|},$$

where $\bar{x}$ is the restored image and x is the original one, see e.g. [26, Appendix 3]. For our numerical experiments, we start the iterations with the degraded images. Therefore, the initial value of SNR is 14.75dB for the chart image and 19.59dB for the house image.

We first investigate the performance of Algorithm 2 with different choices of the relaxation factor γ, and thus verify our theoretical assertion of choosing aggressive values. In Figure 2, we visualize the evaluation of the restored SNR values with respect to iterations when the proposed Algorithm 2 with different values of γ is implemented on the chart image. The preference of aggressive values of γ close to 2 are thus verified by these curves.

The curves in Figure 2 also show that the restored SNR value by Algorithm 2 turns to be stable after about 50 iterations, and the improvement over this stable value is very little even if the iteration keeps running. Thus, we can regard the restored stable SNR value as the asymptotically optimal value, or equivalently, the effectiveness, of a method. Recall that the capability of achieving a higher SNR value reflects a better quality of the restored image for a method. Therefore, for a method, the higher the asymptotically optimal SNR value is, the better the method is. As shown in the convergence analysis, for an iterate v^{k+1} generated by Algorithm 2, its proximity to $\mathcal{V}^*$ can be measured by $\|v^k - \tilde{v}^k\|_M$. Recall $B = -I$ in (6.6). Therefore, we propose to adopt the following stopping criterion for (6.4)

$$\mathrm{Tol} := \max\{\beta\|\tilde{y}^k - y^k\|^2, \|\tilde{\lambda}^k - \lambda^k\|^2/\beta\} < 0.5. \tag{6.8}$$

Empirically, it is reliable to regard that the asymptotically optimal SNR value of a method is approached when (6.8) is satisfied. In Table 2, for each tested method, we report the number of iterations ("Iter") and the computing time in seconds ("CPU") when the criterion (6.8) is satisfied; and the achieved SNR value as well. Finally, we show the restored images by the ADM and the proposed Algorithm 2 in Figure 3.

These preliminary results show clearly that both ADM and Algorithm 2 are equally effective to restore the images via the model (6.4), as they are capable of restoring images with the same (or slightly different) SNR values. On the other hand, Algorithm 2 with an aggressive relaxation factor outperforms the ADM obviously in terms of the restoration speed. Overall, this example further verifies our theoretical assertions: 1) the customized PPA (3.5) is as efficient as the ADM (2.7); 2) the relaxation factor γ in (3.7) can accelerate the convergence of the customized PPA easily and aggressive values close to 2 are often preferred empirically.

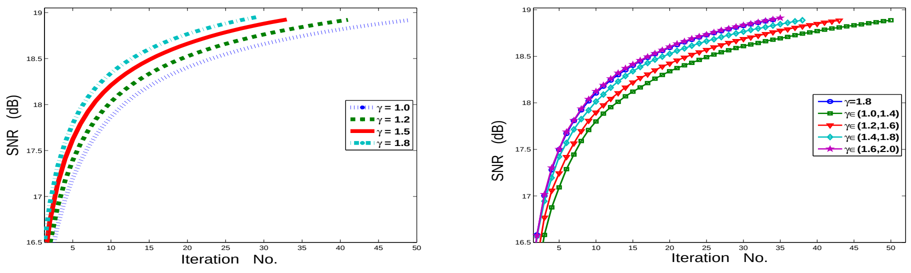

Figure 2: Performances of Algorithm 2 with different values of γ for chart.tif. Left: fixed γ. Right: random generated γ.

Table 2: Numerical comparison of ADM and Algorithm 2 for TV-l^2 image restoration.

	chart				house			
	ADM (2.7)	Algo.2 $\gamma = 1$	Algo.2 $\gamma = 1.5$	Algo.2 $\gamma = 1.8$	ADM (2.7)	Algo.2 $\gamma = 1$	Algo.2 $\gamma = 1.5$	Algo.2 $\gamma = 1.8$
Iter	74	74	50	42	57	56	38	33
CPU	2.32	2.29	1.53	1.29	2.79	2.75	1.81	1.63
SNR	19.01	19.01	19.02	19.02	22.11	22.11	22.12	22.12

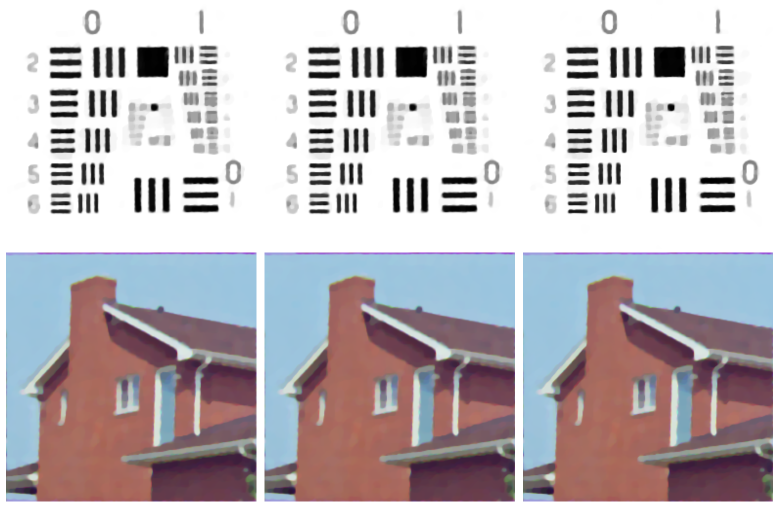

Figure 3: Restored images. From left column to right column: ADM(2.7), Algorithm 2 with $\gamma = 1$, Algorithm 2 with $\gamma = 1.8$.

7 Conclusions

This paper takes a fresh look at the direct application of the proximal point algorithm (PPA) to the separable convex programming with linear constraints, and proposes a customized PPA by taking advantage of the separable structure. It is shown by numerical experiments that the customized PPA is numerically competitive with the alternating direction method (ADM) which is the application of the PPA to the corresponding dual problem. The customized PPA and ADM enjoy the same effectiveness of exploiting the separable structure, the same easiness of algorithmic implementation and the same numerical efficiency. Thus, we reach the conclusion that the PPA can be equally effective for separable convex programming no matter it is applied to the primal or dual problem. This new study from the PPA perspective also enables us to accelerate the customized PPA easily by a simple over-relaxation effort, as our convincing numerical results demonstrate. Therefore, we reach another conclusion that for the broad spectrum of applications where ADM is impressively efficient, the relaxed customized PPA is a strongly recommended substitute to ADM as it outperforms ADM considerately while with almost no additional cost.

References

[1] M. Afonso, J. Bioucas-Dias and M. Figueiredo, *Fast image recovery using variable splitting and constrained optimization*, IEEE Trans. Image Process., 19 (2010), pp. 2345–2356.

[2] S. Becker, J. Bobin and E. Candès, *NESTA: A fast and accurate first-order method for sparse recovery*, SIAM J. Imaging Sci, 4(1)(2009), pp. 1-39.

[3] D. P. Bertsekas, *Constrained Optimization and Lagrange Multiplier Methods*, Academic Press, Newy York, 1982.

[4] E. Blum and W. Oettli, Mathematische Optimierung, Econometrics and Operations Research XX, Springer Verlag, 1975.

[5] R. Borsdorf and N. J. Higham, *A preconditioned Newton algorithm for the nearest correlation matrix*, IMA J. Numer. Anal., 30(1) (2010), pp. 94–107.

[6] S. Boyd, N. Parikh, E. Chu, B. Peleato and J. Eckstein, *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Found. Trends Mach. Learn., 3(2010), pp. 1–122.

[7] S. Boyd and L. Xiao, *Least-squares covariance matrix adjustment*, SIAM J. Matrix Anal. Appl., 27(2005), pp. 532-546.

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

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

[10] E. Esser, *Applications of Lagrangian-based alternating direction methods and connections to split Bregman*, UCLA CAM Report 09-31, 2009.

[11] F. Facchinei and J. S. Pang, *Finite-dimensional Variational Inequalities and Complementarity Problems. Vol. I*, Springer Series in Operations Research. Springer-Verlag, New York, 2003.

[12] D. Gabay and B. Mercier, *A dual algorithm for the solution of nonlinear variational problems via finite element approximation*, Computers and Mathematics with Applications, 2(1) (1976), pp. 17–40.

[13] E. G. Gol'shtein and N. V. Tret'yakov, *Modified Lagrangian in convex programming and their generalizations*, Math. Program. Study, 10 (1979), pp. 86–97.

[14] P. C. Hansen, J. G. Nagy and D. P. O'Leary, *Deblurring Images: Matrices, Spectra, and Filtering*, SIAM, Philadelphia, 2006.

[15] M. R. Hestenes, *Multiplier and gradient methods*, J. Optim. Theory Appl., 4 (1969), pp. 303–320.

[16] B. S. He, M. H. Xu and X. M. Yuan, *Solving large-scale least squares covariance matrix problems by alternating direction methods*, SIAM J. Matrix Anal. Appl., 32(2011), pp. 136–152.

[17] B. S. He and X. M. Yuan, *A contraction method with implementable proximal regularization for linearly constrained convex programming*, submitted, 2010.

[18] B. S. He and X. M. Yuan, *On the $O(1/t)$ convergence rate of the alternating direction method*, submitted, 2011.

[19] N. J. Higham, *Computing the nearest correlation matrix—a problem from finance*, IMA J. Numer. Anal., 22(3) (2002), pp. 329–343.

[20] J. Malick, *A dual approach to semidefinite least-squares problems*, SIAM J. Matrix Anal. Appl., 26(2004), pp. 272-284.

[21] B. Martinet, *Regularision d'inéquations variationnelles par approximations successive*, Revue Francaise d'Automatique et Informatique Recherche Opérationnelle, 126 (1970), pp. 154–159.

[22] A. Nemirovski, *Prox-method with rate of convergence $O(1/t)$ for variational inequalities with Lipschitz continuous monotone operators and smooth convex-concave saddle point problems*, SIAM J. Optim., 15(1)(2004), pp. 229–251.

[23] M. K. Ng, R. H. Chan and W. C. Tang, *A fast algorithm for deblurring models with Neumann boundary conditions*, SIAM J. Sci. Comput., 21(3)(1999), pp. 851–866.

[24] M. K. Ng, P. A. Weiss and X. M. Yuan, *Solving constrained total-variation problems via alternating direction methods*, SIAM J. Sci. Comput., 32(5)(2010), pp. 2710–2736.

[25] M. J. D. Powell, *A method for nonlinear constraints in minimization problems*, In Optimization edited by R. Fletcher, pp. 283–298, Academic Press, New York, 1969.

[26] W. K. Pratt, *Digital Image Processing: PIKS Inside*, 3rd Edition, John Wiley & Sons, Inc, 2001.

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

[28] R.T. Rockafellar, *Monotone operators and the proximal point algorithm*, SIAM J. Control Optim., 14 (1976), pp. 877–898.

[29] L. Rudin, S. Osher and E. Fatemi, *Nonlinear total variation based noise removal algorithms*, Phys. D., 60 (1992), pp. 259–268.

[30] J. Sun and S. Zhang, *A modified alternating direction method for convex quadratically constrained quadratic semidefinite programs*, European J. Oper. Res., 207 (2010), pp. 1210–1220.

[31] M. Tao and X. M. Yuan, *Recovering low-rank and sparse components of matrices from incomplete and noisy observations*, SIAM J. Optim, 21(1) (2011), pp. 57–81.

[32] P. Tseng, *On accelerated proximal gradient methods for convex-concave optimization*, manuscript, 2008.

[33] E. van den Berg and M. P. Friedlander, *Probing the Pareto frontier for basis pursuit solutions*, SIAM J. Sci. Comp., 31 (2008), pp. 890–912.

[34] Y. Wang, J. Yang, W. Yin and Yin Zhang, *A new alternating minimization algorithm for total variation image reconstruction*, SIAM J. Imaging Sci., 1(3) (2008), pp. 248–272.

[35] C. H. Ye and X. M. Yuan, *A descent method for structured monotone variational inequalities*, Optim. Methods Softw., 22(2)(2007), pp. 329–338.