Hindawi Publishing Corporation
Abstract and Applied Analysis
Volume 2013, Article ID 492305, 13 pages
http://dx.doi.org/10.1155/2013/492305

Research Article

A General Self-Adaptive Relaxed-PPA Method for Convex Programming with Linear Constraints

Xiaoling Fu

Institute of Systems Engineering, Southeast University, Nanjing 210096, China

Correspondence should be addressed to Xiaoling Fu; fuxlnju@hotmail.com

Received 27 June 2013; Accepted 21 July 2013

Academic Editor: Abdellah Bnouhachem

We present an efficient method for solving linearly constrained convex programming. Our algorithmic framework employs an implementable proximal step by a slight relaxation to the subproblem of proximal point algorithm (PPA). In particular, the stepsize choice condition of our algorithm is weaker than some elegant PPA-type methods. This condition is flexible and effective. Self-adaptive strategies are proposed to improve the convergence in practice. We theoretically show under mild conditions that our method converges in a global sense. Finally, we discuss applications and perform numerical experiments which confirm the efficiency of the proposed method. Comparisons of our method with some state-of-the-art algorithms are also provided.

1. Introduction

In this paper, we consider the following generic convex programming:

$$\min\{\theta(x) \mid Ax = b\,(\text{or } Ax \geq b),\ x \in \mathcal{X}\}, \tag{1}$$

where $\theta(x) : \Re^n \to \Re$ is a convex (not necessary smooth) function, $A \in \Re^{m\times n}$, $b \in \Re^m$, and $\mathcal{X} \subset \Re^n$ is a closed convex set. Problem (1) generalizes a wide range of problems that frequently arise in signal and image processing and reconstruction, mechanics, statistics, operations research, and other fields, for example, basis pursuit [1–4], nearest correlation matrix [5–7], matrix completion problem [2, 3, 8–10], and so forth. Before we begin, some assumptions should be presented for problem (1).

Assumption 1. The solution set of (1) is denoted by $\mathcal{X}^*$, and it is assumed to be nonempty.

Assumption 2. The objective function is simple. This means that, for a given constant ϱ, the following proximal problem admits a closed-form solution or can be solved efficiently with high precision:

$$\min_{x\in\mathcal{X}} \theta(x) + \frac{\varrho}{2}\|x - a\|^2, \tag{2}$$

where a is any given vector. At first sight, this assumption seems to be quite restrictive, but this is indeed for many problems in practice. For example, nuclear norm function in matrix completion problem, l_1-norm function in basis pursuit problem, and so forth.

Many fundamental methods have been developed over the past decades to solve problem (1). Proximal point algorithm (PPA) is one of the leading approaches for solving convex optimization problems. It is earlier used for regularized linear equations and has been applied to convex optimization by Martinet [11]. There are some significant theoretical achievements [12–19] in the field of PPA for convex optimization and monotone variational inequalities (VIs). Nowadays, it is still the object of intensive investigation [20] and leads to a variety of primal and dual methods. Common to PPA and its variants is the difficulty of their subproblems; this restricts the practical interest. Augmented Lagrangian method (ALM) [21] is a powerful method for linearly constrained problems. It can be regarded as a variant of PPA applied to the dual problem of (1). However, with the additional regularized term $\|Ax - b\|^2$, its subproblems require an inverse operator of the form $(I+(1/s)A^TA)^{-1}$ which is hard to implement in some cases. Particularly, A^TA is general or large scale, so the computation of inverse operator may fail. Hence,

ALM is not sufficiently competitive when the objective function $\theta(x)$ is "simple." Extragradient method (EGM) [22] is a practical method for (1) which employs the information of current iteration. In fact, EGM is an explicit type method and requires two calls to the gradient per iteration; therefore, it might suffer slow convergence. Recently, He and Yuan [23] proposed a contraction method based on PPA (PPA-CM) to solve (1), which is elegant and simple. Inspired by PPA-CM, a Lagrangian PPA-based (LPPA) contraction method is presented in [24] which employs an asymmetrical proximal term [25]. These two PPA-based methods have nice convergence properties that are similar in many ways to PPA, but they both require a quite restrictive condition for convergence and are sensitive to the initial choice of stepsizes.

In this paper, we focus on development of PPA-type method for solving (1). Based on LPPA, we propose a general self-adaptive relaxed-PPA method (SRPPA) which is simple yet efficient. Our algorithm capitalizes certain features of PPA, hence, we term it relaxed-PPA. The proposed algorithm has several nice fronts. First, our method is a PPA-type method with asymmetrical linear term, which is clearly a different nature to classical PPA. It relaxes the jointly structure of subproblem to a tractable one. Second, we provide two simple search directions for new iterate. Third, the stepsize choice is flexible, and the condition for convergence guarantee is weaker than both PPA-CM and LPPA. Finally, simple adaptive strategies are employed to choose stepsize, and this appealing aspect is significantly important in practice. We also demonstrate that our method is relevant for various applications whose practical success is made possible by our efficient algorithm.

This paper is organized as follows. In Section 2, we provide some notations and preliminaries which are useful for subsequent analysis. In Section 3, we review some related works. The general relaxed-PPA and its variant are formally presented in Section 4. Self-adaptive strategies to choose stepsize are also described. Next, in Section 5, the convergence analysis is provided. In Section 6, we present some concrete applications of (1) and elaborate on the implementation of our method; preliminary numerical results are also reported to verify the efficiency of our proposed method. Finally, in Section 7, we conclude the paper with a discussion about the future research directions.

2. Preliminaries

In this section, we first establish some important notations used throughout this paper. Then, we describe the variational inequality formulation of (1) which is convenient for the convergence analysis.

$\partial\theta(x)$ denotes the subdifferential set of the convex function $\theta(x)$:

$$\partial\theta(x) := \left\{d \in \Re^n \mid \theta(y) - \theta(x) \geq d^T(y-x),\ \forall y \in \Re^n\right\}, \tag{3}$$

and $d \in \partial\theta(x)$ is called a subgradient of $\theta(x)$, see [26]. Let $f(x) \in \partial\theta(x)$ and $f(y) \in \partial\theta(y)$, by the convexity of the function θ, we have

$$(x-y)^T(f(x) - f(y)) \geq 0, \quad \forall x, y \in \Re^n, \tag{4}$$

which indicates that the mapping f is monotone.

Now, we show that (1) can be characterized by a variational inequality; see, for example, [27]. By attaching a Lagrange multiplier vector $\lambda \in \Lambda$ to the linear constraint $Ax = b$ (or $Ax \geq b$), the Lagrangian function of (1) is

$$L(x,\lambda) = \theta(x) - \lambda^T(Ax - b); \tag{5}$$

here,

$$\Lambda = \begin{cases} \Re^m, & \text{for the equality constraints } Ax = b, \\ \Re^m_+, & \text{for the inequality constraints } Ax \geq b, \end{cases} \tag{6}$$

and $L(x,\lambda)$ is defined on $\mathcal{X} \times \Lambda$. Then, by the optimality condition, we can easily see that (1) amounts to finding a pair of (x^*, λ^*) which satisfies

$$\begin{aligned} &x^* \in \mathcal{X}, \quad (x - x^*)^T\left\{f(x^*) - A^T\lambda^*\right\} \geq 0, \quad \forall x \in \mathcal{X}, \\ &\lambda^* \in \Lambda, \quad (\lambda - \lambda^*)^T(Ax^* - b) \geq 0, \quad \forall \lambda \in \Lambda, \end{aligned} \tag{7}$$

where $f(x^*) \in \partial\theta(x^*)$. Denoting

$$u = \begin{pmatrix} x \\ \lambda \end{pmatrix}, \quad F(u) = \begin{pmatrix} f(x) - A^T\lambda \\ Ax - b \end{pmatrix}, \qquad \Omega = \mathcal{X} \times \Lambda, \tag{8}$$

the system (7) can be characterized by the following variational inequality denoted by VI(Ω, F):

$$u^* \in \Omega, \quad (u - u^*)^T F(u^*) \geq 0, \quad \forall u \in \Omega. \tag{9}$$

Recalling the monotonicity of f, it is easy to get that VI(Ω, F) (9) is monotone. Since the solution set of (1) is assumed to be nonempty, the solution set of VI(Ω, F), denoted by Ω^*, is also nonempty. Our analysis will be built upon this equivalent VI formulation.

3. The Existing Related Methods

There are basically two lines of research for VI(Ω, F) (9), either deal with it by implicit methods that are in general computationally intractable or concentrate on relaxing it with explicit methods. In this section, we first briefly review the well-known classical PPA and EGM. And then, PPA-CM [23] and LPPA [24] are discussed, which will provide motivation for our general self-adaptive relaxed-PPA.

3.1. Classical PPA for the Equivalent Variational Inequality. PPA and its variants are implicit methods which have fast asymptotical convergence rate and produce highly accurate solutions. At each iteration, the subproblem of classical PPA consists of a regularized term, which can be expressed as

follows: given any iterate $u^k = (x^k, \lambda^k)$, find $\tilde{u}^k = (\tilde{x}^k, \tilde{\lambda}^k) \in \Omega$ such that

$$\tilde{u}^k \in \Omega, \quad \left(u - \tilde{u}^k\right)^T \left\{F\left(\tilde{u}^k\right) + r\left(\tilde{u}^k - u^k\right)\right\} \geq 0, \quad \forall u \in \Omega. \tag{10}$$

Then, the update step is taken as follows:

$$u^{k+1} = u^k - \gamma\left(u^k - \tilde{u}^k\right), \quad \gamma \in (0, 2). \tag{11}$$

PPA has a nice convergence property

$$\left\|u^{k+1} - u^*\right\|^2 \leq \left\|u^k - u^*\right\|^2 - \gamma(2 - \gamma)\left\|u^k - \tilde{u}^k\right\|^2. \tag{12}$$

Although classical PPA is conceptually appealing, subproblem (10) presents certain computational challenges. More specifically, primal variable x and dual variable λ are tied together, and their subproblems are treated as a joint problem. In most cases, this joint subproblem may be as difficult as the original problem (9). As a result, PPA is "conceptual" rather than implementable. It lacks capability in exploiting potential decomposable/specific structure of subproblem. Variants of classical PPA have been explored in the literature, in order to solve the proximal subproblem (10), inexactly, see, for example, [14, 15, 17, 19]. Unfortunately, inexact variants take expensive computation for obtaining approximative solutions.

3.2. The Methods Based on the Simplest Relaxation. To overcome the drawbacks of the classical PPA, it is instinctive to relax subproblem (10) to a solvable one. The most straightforward and simplest relaxation is to replace $F(\tilde{u}^k)$ with $F(u^k)$ in the proximal subproblem (10), which amounts to the following subproblem:

$$\tilde{u}^k \in \Omega, \quad \left(u - \tilde{u}^k\right)^T \left\{F\left(u^k\right) + r\left(\tilde{u}^k - u^k\right)\right\} \geq 0, \quad \forall u \in \Omega. \tag{13}$$

The solution of the relaxed problem (13) is given by $\tilde{u}^k = P_\Omega[u^k - (1/r)F(u^k)]$. It is clear that methods with such relaxation are explicit type methods. However, $\tilde{u}^k$ cannot be accepted directly as the new iterate. Using the terminology "*predictor-corrector*," such point can be viewed as a *predictor*. Here, we list two simple methods which employ *predictor* $\tilde{u}^k$ to obtain *corrector* as the new iterate.

(i) The extragradient method (EGM) updates the new iterate (corrector) by

$$u^{k+1} = P_\Omega\left[u^k - \frac{1}{r}F\left(\tilde{u}^k\right)\right]. \tag{14}$$

(ii) The projection and contraction methods (PCM) [28–30] perform update as follows:

$$u^{k+1} = u^k - \gamma\alpha_k^* d^k \quad \text{or} \quad u^{k+1} = P_\Omega\left[u^k - \frac{\gamma\alpha_k^*}{r}F\left(\tilde{u}^k\right)\right], \tag{15}$$

where

$$d^k = \left(u^k - \tilde{u}^k\right) - \frac{1}{r}\left(F\left(u^k\right) - F\left(\tilde{u}^k\right)\right),$$
$$\alpha_k^* = \frac{\left(u^k - \tilde{u}^k\right)^T d^k}{\left\|d^k\right\|^2}. \tag{16}$$

The sequence $\{u^k\}$ generated by the above mentioned EGM or PCM satisfies

$$\left\|u^{k+1} - u^*\right\|^2 \leq \left\|u^k - u^*\right\|^2 - c\left\|u^k - \tilde{u}^k\right\|^2, \quad c > 0, \tag{17}$$

which is similar to PPA. Both EGM and PCM use the simplest relaxation to obtain $\tilde{u}^k$ in kth iteration, hence are computationally practical. These methods have appealing practical aspects; however, such simplest relaxation does not exploit the inner structure of VI(Ω, F) (9). This observation prompts the need for dedicated relaxations.

3.3. PPA-Type Contraction Method. The algorithms that are closely related to ours are PPA-CM [23] and LPPA [24]. The PPA-CM obtains the predictor $\tilde{u}^k$ by solving the following subproblem: find $(\tilde{x}^k, \tilde{\lambda}^k) \in \Omega$ such that

$$\tilde{u}^k \in \Omega, \quad \left(u - \tilde{u}^k\right)^T \left\{F\left(\tilde{u}^k\right) + S\left(\tilde{u}^k - u^k\right)\right\} \geq 0, \quad \forall u \in \Omega, \tag{18}$$

where

$$S = \begin{pmatrix} rI_n & -A^T \\ -A & sI_m \end{pmatrix}. \tag{19}$$

And perform the update

$$u^{k+1} = u^k - \gamma\left(u^k - \tilde{u}^k\right), \quad \gamma \in (0, 2). \tag{20}$$

The framework of LPPA is as follows:

$$\tilde{u}^k \in \Omega, \quad \left(u - \tilde{u}^k\right)^T \left\{F\left(\tilde{u}^k\right) + M\left(\tilde{u}^k - u^k\right)\right\} \geq 0, \quad \forall u \in \Omega, \tag{21}$$

where

$$M = \begin{pmatrix} rI_n & A^T \\ 0 & sI_m \end{pmatrix}. \tag{22}$$

And the new iterate is defined by

$$u^{k+1} = u^k - \gamma\alpha M\left(u^k - \tilde{u}^k\right), \quad \gamma \in (0, 2). \tag{23}$$

Both procedures are simple and can solve subproblem efficiently; but their nice convergence results require a quite restrictive condition, that is; $rs > \|A^T A\|$ in PPA-CM and $rs > (1/2)\|A^T A\|$ in LPPA, respectively. The stepsizes r, s are directly determined by such condition; hence, it is important to estimate $\|A^T A\|$. Overestimation may lead to poor stepsizes and slow convergence, while underestimation may result in divergence. In addition, they are both quite sensitive to the choice of r, s. To overcome those drawbacks, we propose a general self-adaptive relaxed-PPA, and as mentioned earlier, it can provide improved guarantee for convergence and has potential progress in the choice of stepsize. Furthermore, self-adaptive strategies are designed to improve performance.

Step 1. Initialization. Let $\gamma \in (0,2)$ and pick $(x^0, \lambda^0) \in \Re^n \times \Lambda$, set $k = 0$.
Step 2. Predictor. Let

$$\tilde{x}^k = \operatorname{Argmin}\left\{\theta(x) + \frac{r}{2}\left\|x - \left[x^k + \frac{1}{r}A^T\lambda^k\right]\right\|^2 \mid x \in \mathcal{X}\right\}, \quad (*)$$

and

$$\tilde{\lambda}^k = P_\Lambda\left[\lambda^k - \frac{1}{s}(A\tilde{x}^k - b)\right]. \quad (**)$$

Step 3. If the stepsizes r and s are chosen satisfying

$$\varphi(u^k, \tilde{u}^k) \geq \frac{1}{4}\left\|d(u^k, \tilde{u}^k)\right\|_G^2, \quad (\#)$$

then go to Step 4. Otherwise, increase r, s and go back to Step 2.
Step 4. Corrector and updating

$$u^{k+1} = u^k - \alpha_k G^{-1} M(u^k - \tilde{u}^k). \quad (\$)$$

Step 5. Adjustment

$$(r,s) = \begin{cases} \left(\frac{r}{2}, \frac{s}{2}\right), & \text{if } \varphi(u^k, \tilde{u}^k) \geq \kappa * \left\|d(u^k, \tilde{u}^k)\right\|_G^2; \\ (r,s), & \text{otherwise.} \end{cases} \quad \text{Here } \kappa > 4.$$

Set $k := k + 1$.

Algorithm 1: General primal-dual relaxed-PPA method.

Step 1. Initialization. Let $\gamma \in (0,2)$ and pick $(x^0, \lambda^0) \in \Re^n \times \Lambda$, set $k = 0$.
Step 2. Predictor. Let

$$\tilde{\lambda}^k = P_\Lambda\left[\lambda^k - \frac{1}{s}(Ax^k - b)\right], \quad (\dagger)$$

and

$$\tilde{x}^k = \operatorname{Argmin}\left\{\theta(x) + \frac{r}{2}\left\|x - \left[x^k + \frac{1}{r}A^T\tilde{\lambda}^k\right]\right\|^2 \mid x \in \mathcal{X}\right\}. \quad (\dagger\dagger)$$

Step 3. If the stepsize r and s are chosen satisfying

$$\varphi\left(u^k, \tilde{u}^k\right) \geq \frac{1}{4}\left\|d\left(u^k, \tilde{u}^k\right)\right\|_G^2, \quad (\ddagger)$$

then go to Step 4. Otherwise, increase r, s and go back to Step 2.
Step 4. Corrector and updating

$$u^{k+1} = u^k - \alpha_k G^{-1} M(u^k - \tilde{u}^k). \quad (\wedge)$$

Step 5. Adjustment

$$(r,s) = \begin{cases} \left(\frac{r}{2}, \frac{s}{2}\right), & \text{if } \varphi\left(u^k, \tilde{u}^k\right) \geq \kappa * \left\|d\left(u^k, \tilde{u}^k\right)\right\|_G^2; \\ (r,s), & \text{otherwise.} \end{cases} \quad \text{Here } \kappa > 4.$$

Set $k := k + 1$.

Algorithm 2: General dual-primal relaxed-PPA method.

4. The General Self-Adaptive Relaxed PPA-Method

In this section, we weave together the ideas of the previous section to present general self-adaptive relaxed-PPA method (SRPPA) which is mostly inspired by LPPA [24]. At first sight, the predictor applied in SRPPA is much the same as LPPA, but the stepsize choice condition for convergence is quite different; moreover, we prove that it is weaker than LPPA. Self-adaptive strategies are elaborately designed to ensure the robustness of our algorithm. Two simple yet efficient constructions of new iterate are also presented which will provide some inspirations for designing various search directions.

4.1. General Relaxed-PPA Method. The general primal-dual relaxed-PPA method with implementable structure for (1) is summarized in Algorithm 1. Note that the order of x and λ can be changed to obtain a variant, which is summarized in Algorithm 2. Our relaxed-PPA is intended to blend the implementable properties of EGM (or PCM) with the fast convergence performance of PPA. Now, it is helpful to introduce additional notations that will be used in the rest of this paper. Let G be a positive symmetry definite matrix (we will specify it later),

$$M = \begin{pmatrix} rI_n & A^T \\ 0 & sI_m \end{pmatrix}, \qquad H = \begin{pmatrix} rI_n & 0 \\ 0 & sI_m \end{pmatrix}, \quad (24)$$

$$d\left(u^k, \tilde{u}^k\right) = G^{-1}M\left(u^k - \tilde{u}^k\right), \quad (25)$$

$$\varphi\left(u^k, \tilde{u}^k\right) = \left(u^k - \tilde{u}^k\right)^T M\left(u^k - \tilde{u}^k\right). \quad (26)$$

The relaxed-PPA described here involves two steps. First, we solve the relaxed subproblem $(*)$, $(**)$ to obtain predictor, which is nice and efficient for the nature of the problem under study. Note that the x-predictor step $(*)$ involves minimizing θ plus a convex quadratic function, and under Assumption 2, it can be efficiently solved or it admits a closed form solution. And then, λ-predictor step $(**)$ is just a projection onto Λ which is tractable and computationally efficient. It is clear that the prediction step employs a Gauss-Seidel manner to update information efficiently. The correction step ($) only involves matrix-vector multiplication which is very simple and straightforward.

Remark 3. We first make some insight into the correction step in Algorithm 1. The obtained $\tilde{u}^k$ plays no direct role as the new iterate. Instead, we need some kind of "corrector" defined in ($). Although matrix G in ($) is just a required positive symmetry definite, our goal here is to fully integrate the information of u^k and $\tilde{u}^k$ to construct effective, simple search direction $G^{-1}M(u^k - \tilde{u}^k)$ for the corrector. Based on this consideration, we elaborately provide two simple choices of G.

Case 1. It is natural to set $G = H$ to induce a simple update form. Then, it is easy to get that

$$\begin{pmatrix} x^{k+1} \\ \lambda^{k+1} \end{pmatrix} = \begin{pmatrix} x^k \\ \lambda^k \end{pmatrix} - \alpha_k \begin{pmatrix} x^k - \tilde{x}^k + \frac{1}{r} A^T \left(\lambda^k - \tilde{\lambda}^k\right) \\ \lambda^k - \tilde{\lambda}^k \end{pmatrix}. \quad (27)$$

Case 2. Let $G = MH^{-1}M^T$. This case is a little less intuitive, but it can lead to a simple update form as well as Case 1. The underlying derivation is a little more complicate. Applying G in ($), we get

$$\begin{aligned} u^{k+1} &= u^k - \alpha_k M^{-T} H M^{-1} M \left(u^k - \tilde{u}^k\right) \\ &= u^k - \alpha_k M^{-T} H \left(u^k - \tilde{u}^k\right). \end{aligned} \quad (28)$$

Recalling M is a lower triangular matrix, by the fact that its inverse is also a lower triangular, we have

$$G^{-1}M = M^{-T}H = \begin{pmatrix} I_n & 0 \\ -\frac{A}{s} & I_m \end{pmatrix}. \quad (29)$$

Plugging the previous relationship to (31), we have

$$\begin{pmatrix} x^{k+1} \\ \lambda^{k+1} \end{pmatrix} = \begin{pmatrix} x^k \\ \lambda^k \end{pmatrix} - \alpha_k \begin{pmatrix} x^k - \tilde{x}^k \\ \lambda^k - \tilde{\lambda}^k - \frac{1}{s} A \left(x^k - \tilde{x}^k\right) \end{pmatrix}. \quad (30)$$

In fact, this is a scheme of Gaussian back substitution.

Both cases only involve one matrix-vector multiplication which makes the update form simple. And the computational cost is usually inexpensive.

Remark 4. We now study the subproblem described in $(*)$, $(**)$, and the stepsize choice condition (#). For easy analysis, we characterize $(*)$, $(**)$ as the following VI formulation.

Find $(\tilde{x}^k, \tilde{\lambda}^k) \in \Omega$ such that

$$\begin{aligned} &\begin{pmatrix} x - \tilde{x}^k \\ \lambda - \tilde{\lambda}^k \end{pmatrix}^T \left\{ \begin{pmatrix} f\left(\tilde{x}^k\right) - A^T \tilde{\lambda}^k \\ A\tilde{x}^k - b \end{pmatrix} + \begin{pmatrix} rI_n & A^T \\ 0 & sI_m \end{pmatrix} \begin{pmatrix} \tilde{x}^k - x^k \\ \tilde{\lambda}^k - \lambda^k \end{pmatrix} \right\} \\ &\quad \geq 0, \quad \forall \begin{pmatrix} x \\ \lambda \end{pmatrix} \in \Omega, \end{aligned} \quad (31)$$

and its compact form

$$\begin{aligned} &\tilde{u}^k \in \Omega, \quad \left(u - \tilde{u}^k\right)^T \left\{ F\left(\tilde{u}^k\right) + M\left(\tilde{u}^k - u^k\right) \right\} \geq 0, \\ &\qquad \forall u \in \Omega. \end{aligned} \quad (32)$$

We observe that subproblem (32) is similar to (10) in PPA, except for the construction of asymmetry matrix M. As mentioned before, (32) is the same as the prediction subproblem in [24]. Even though they are closely related, the stepsize choice here is quite different. We provide more specific and weaker condition for stepsize r, s. It is clear that condition (#) does not need prior knowledge of matrix A. Furthermore, it only involves matrix-vector multiplication, and so, it is easy to verify, and it is amenable to large-scale A. If r, s fail to meet this convergence condition (#), one should appropriately increase r, s. In the following subsection, we will elaborate on the self-adaptive strategies to increase the stepsizes. At this point, condition (#) may be seen somewhat unmotivated. Some insight into this will be provided later, as we proceed with the convergence analysis. The convergence condition in [24] has a quite different feature: r, s satisfy

$$rs \geq \frac{1}{2} \left\| A^T A \right\|. \quad (33)$$

It is stronger than condition (#). The following lemma is devoted to the proof of this result.

Lemma 5. *Let $\{u^k\}$ be the sequence generated by Algorithm 1, H, $d(u^k, \tilde{u}^k)$, and $\varphi(u^k, \tilde{u}^k)$ defined in* (24), (25), *and* (26), *respectively. Suppose that condition* (33) *is satisfied. Then, condition (#) holds immediately.*

Proof. Note that

$$\begin{aligned} \varphi\left(u^k, \tilde{u}^k\right) &= r\left\|x^k - \tilde{x}^k\right\|^2 + s\left\|\lambda^k - \tilde{\lambda}^k\right\|^2 \\ &\quad + \left(\lambda^k - \tilde{\lambda}^k\right)^T A \left(x^k - \tilde{x}^k\right). \end{aligned} \quad (34)$$

Recall that matrix G described in Algorithm 1 can be designed in two different cases.

Case 1. If $G = H$, we immediately have

$$\begin{aligned} \left\|d\left(u^k, \tilde{u}^k\right)\right\|_G^2 &= \left\|u^k - \tilde{u}^k\right\|_H^2 + \frac{1}{r}\left\|A^T(\lambda^k - \tilde{\lambda}^k)\right\|^2 \\ &\quad + 2\left(\lambda^k - \tilde{\lambda}^k\right)^T A \left(x^k - \tilde{x}^k\right). \end{aligned} \quad (35)$$

According to Cauchy-Schwarz inequality, we get

$$\begin{aligned}
&\varphi\left(u^k,\tilde{u}^k\right)-\frac{1}{4}\left\|d\left(u^k,\tilde{u}^k\right)\right\|_G^2\\
&\quad=\frac{3}{4}\left\|u^k-\tilde{u}^k\right\|_H^2-\frac{1}{4r}\left\|A^T\left(\lambda^k-\tilde{\lambda}^k\right)\right\|^2\\
&\qquad+\frac{1}{2}\left(\lambda^k-\tilde{\lambda}^k\right)^TA\left(x^k-\tilde{x}^k\right)\\
&\quad\geq\frac{3}{4}\left\|u^k-\tilde{u}^k\right\|_H^2-\frac{1}{4r}\left\|A^T(\lambda^k-\tilde{\lambda}^k)\right\|^2\\
&\qquad-\frac{1}{4}\left(2r\left\|x^k-\tilde{x}^k\right\|^2+\frac{1}{2r}\left\|A^T\left(\lambda^k-\tilde{\lambda}^k\right)\right\|^2\right)\\
&\quad=\frac{r}{4}\left\|x^k-\tilde{x}^k\right\|^2+\frac{3s}{4}\left\|\lambda^k-\tilde{\lambda}^k\right\|^2\\
&\qquad-\frac{3}{8r}\left\|A^T(\lambda^k-\tilde{\lambda}^k)\right\|^2\geq\frac{r}{4}\left\|x^k-\tilde{x}^k\right\|^2.
\end{aligned}\tag{36}$$

The last inequality follows directly from condition (33).

Case 2. If $G=MH^{-1}M^T$, by the definition of $d(u^k,\tilde{u}^k)$, we obtain

$$\begin{aligned}
\left\|d\left(u^k,\tilde{u}^k\right)\right\|_G^2&=\left(u^k-\tilde{u}^k\right)^TM^TG^{-1}M\left(u^k-\tilde{u}^k\right)\\
&=\left\|u^k-\tilde{u}^k\right\|_H^2.
\end{aligned}\tag{37}$$

Then, we get

$$\begin{aligned}
&\varphi\left(u^k,\tilde{u}^k\right)-\frac{1}{4}\left\|d\left(u^k,\tilde{u}^k\right)\right\|_G^2\\
&\quad=\frac{3}{4}\left\|u^k-\tilde{u}^k\right\|_H^2+\left(\lambda^k-\tilde{\lambda}^k\right)^TA\left(x^k-\tilde{x}^k\right)\\
&\quad\geq\frac{3}{4}\left\|u^k-\tilde{u}^k\right\|_H^2-\frac{1}{2}\frac{4r}{3}\left\|x^k-\tilde{x}^k\right\|^2\\
&\qquad-\frac{1}{2}\frac{3}{4r}\left\|A^T\left(\lambda^k-\tilde{\lambda}^k\right)\right\|^2\\
&\quad=\frac{r}{12}\left\|x^k-\tilde{x}^k\right\|^2+\frac{3s}{4}\left\|\lambda^k-\tilde{\lambda}^k\right\|^2-\frac{3}{8r}\left\|A^T(\lambda^k-\tilde{\lambda}^k)\right\|^2\\
&\quad\geq\frac{r}{12}\left\|x^k-\tilde{x}^k\right\|^2.
\end{aligned}\tag{38}$$

The first inequality follows from the Cauchy-Schwarz inequality, and the last one follows directly from condition (33).

Note that, in both cases, we have that condition (#) holds if $rs\geq(1/2)\|A^TA\|$. □

Condition (33) is not only stronger than Condition (#), but it also requires that matrix M is positive semidefinite, while condition (#) does not. Furthermore, condition (33) may require the explicit expression of A or knowledge of $\|A^TA\|$. Despite these drawbacks, condition (33) is appealing to the problems in which $\|A^TA\|$ is known beforehand or easy to compute/obtain. For instance, A is small scale, an identity matrix or a projection operator, and so forth. It is clear that both condition (#) and (33) are more flexible than the one in PPA-CM [23]. The most aggressive condition (#) may lead to further improvement in stepsize choice. Moreover, it is worthwhile to notice that condition (#) is elegantly designed and provides $\varphi(u^k,\tilde{u}^k)$ with favourable property. In fact, for general matrix G, condition (#) also can guarantee convergence.

Remark 6. The update stepsize α_k plays an important role here. In fact, it can be regarded as an optimal stepsize which will be illustrated in the following section.

Remark 7. We should restrict the adjustment in Step 5 of Algorithm 1 within a limited number to avoid divergence.

In Algorithm 1, we carry out the x-predictor before performing λ-predictor. The roles of x and λ are symmetric; hence, sweeping the order will not break the Gauss-Seidel structure. We switch x and λ and obtain a variant of relaxed-PPA with the order of the x-predictor step and λ-predictor step reversed. This variant is illustrated in Algorithm 2. However, there is no a priori information to know which algorithm is superior. Here, we let

$$M=\begin{pmatrix}rI_n & 0\\ -A & sI_m\end{pmatrix}.\tag{39}$$

4.2. Adaptive Enhancements. Both PPA-CM and LPPA employ fixed stepsizes r,s. Experiments reveal that they will suffer slow convergence when stepsizes r,s are chosen inappropriately. A natural question is, how to choose the proper initial stepsizes r,s. Here, we propose self-adaptive strategies with the goal of improving the convergence in practice, as well as making performance less dependent on the initial choice of stepsizes. Our strategies dynamically incorporate the information of the current iteration to perform more informative choice of stepsizes for the next iteration [31]. When doing so, the algorithm will be adaptive and free from the initial choice. Denote

$$\begin{pmatrix}d_x^k\\ d_\lambda^k\end{pmatrix}=\begin{pmatrix}\left(x^k-\tilde{x}^k\right)+\dfrac{1}{r}A^T\left(\lambda^k-\tilde{\lambda}^k\right)\\ \lambda^k-\tilde{\lambda}^k\end{pmatrix},\tag{40}$$

and then, (31) can be rewritten as

$$\begin{pmatrix}x-\tilde{x}^k\\ \lambda-\tilde{\lambda}^k\end{pmatrix}^T\left\{\begin{pmatrix}f\left(\tilde{x}^k\right)-A^T\tilde{\lambda}^k\\ A\tilde{x}^k-b\end{pmatrix}-\begin{pmatrix}rI_n & 0\\ 0 & sI_m\end{pmatrix}\begin{pmatrix}d_x^k\\ d_\lambda^k\end{pmatrix}\right\}\geq 0,\quad\forall\begin{pmatrix}x\\ \lambda\end{pmatrix}\in\Omega.\tag{41}$$

Under H-norm, the quantity d_x^k can be viewed as a residual for the dual feasibility condition, and d_λ^k can be viewed as a primal residual. These two residuals converge to zero as relaxed-PPA proceeds. Note that

$$\begin{aligned}
\left\|d_x^k\right\|_r^2&=r\left\|x^k-\tilde{x}^k\right\|^2+2\left(x^k-\tilde{x}^k\right)^TA^T\left(\lambda^k-\tilde{\lambda}^k\right)\\
&\quad+\frac{1}{r}\left\|A^T\left(\lambda^k-\tilde{\lambda}^k\right)\right\|^2,\\
\left\|d_\lambda^k\right\|_s^2&=s\left\|\lambda^k-\tilde{\lambda}^k\right\|^2.
\end{aligned}\tag{42}$$

If $\left\|d_x^k\right\|_r^2 \geq \tau_1 \left\|d_\lambda^k\right\|_s^2$
$r := r; \quad s := s * 2;$
else if $\tau_2 \left\|d_x^k\right\|_r^2 \leq \left\|d_\lambda^k\right\|_s^2$
$r := r * 2; \quad s := s;$
else
$r := r * 1.5; s := s * 1.5.$

Algorithm 3: Adaptation-I.

if $r\left\|x^k - \tilde{x}^k\right\|^2 \geq \tau_1 \left(s\left\|\lambda^k - \tilde{\lambda}^k\right\|^2 + \frac{1}{r}\left\|A^T(\lambda^k - \tilde{\lambda}^k)\right\|^2\right)$
$r := \frac{r}{2}; s = \frac{\mu\left\|AA^T\right\|}{r};$
else if $\tau_2 r\left\|x^k - \tilde{x}^k\right\|^2 \leq \left(s\left\|\lambda^k - \tilde{\lambda}^k\right\|^2 + \frac{1}{r}\left\|A^T(\lambda^k - \tilde{\lambda}^k)\right\|^2\right)$
$s := \frac{s}{2}; r = \frac{\mu\left\|AA^T\right\|}{s};$
else
$r := r; \quad s := s.$

Algorithm 4: Adaptation-II.

And this implies that small values of s tend to reduce the primal residual but have potential to enlarge violations of dual feasibility and tend to produce larger dual residual. This observation motivates us to balance primal and dual residuals. When condition (#) fails, we increase stepsizes r, s properly according to the adaptation shown in Algorithm 3.

Here, $\tau_1 > 1$, $\tau_2 > 1$. This adaptation strategy increases s when the dual residual $\|d_x^k\|_r^2$ appears large compared to the primal residual $\|d_\lambda^k\|_s^2$ and increases r when the dual residual $\|d_x^k\|_r^2$ seems too small relative to the primal residual $\|d_\lambda^k\|_s^2$.

As mentioned, condition (33) is stronger than condition (#). If one chooses condition (33), our RPPA also converges. It must have predetermined stepsizes satisfying $rs = \mu\|A^TA\|$ (here, $\mu \geq 0.5$). However, there is no priority knowledge of the choice of individual r or s. Here, we can also adjust r, s automatically when choosing condition (33). Intuitively, we consider expansion of the entire residual under H-norm:

$$\begin{aligned}&\left\|d_x^k\right\|_r^2 + \left\|d_\lambda^k\right\|_s^2 \\ &\quad = r\left\|x^k - \tilde{x}^k\right\|^2 + s\left\|\lambda^k - \tilde{\lambda}^k\right\|^2 \\ &\qquad + \frac{1}{r}\left\|A^T\left(\lambda^k - \tilde{\lambda}^k\right)\right\|^2 + 2\left(x^k - \tilde{x}^k\right)^T A^T\left(\lambda^k - \tilde{\lambda}^k\right);\end{aligned} \tag{43}$$

there are three terms involving r or s, and we intend to balance these terms. For fixed μ, take $s = (\mu/r)\|AA^T\|$; then

$$s\left\|\lambda^k - \tilde{\lambda}^k\right\|^2 = \frac{\mu}{r}\left\|AA^T\right\|\left\|\lambda^k - \tilde{\lambda}^k\right\|^2. \tag{44}$$

Applying (44) into (43), clearly we have

$$\begin{aligned}&\left\|d_x^k\right\|_r^2 + \left\|d_\lambda^k\right\|_s^2 \\ &\quad = r\left\|x^k - \tilde{x}^k\right\|^2 \\ &\qquad + \frac{1}{r}\left(\mu\left\|AA^T\right\|\left\|\lambda^k - \tilde{\lambda}^k\right\|^2 + \left\|A^T\left(\lambda^k - \tilde{\lambda}^k\right)\right\|^2\right) \\ &\qquad + 2\left(x^k - \tilde{x}^k\right)^T A^T\left(\lambda^k - \tilde{\lambda}^k\right).\end{aligned} \tag{45}$$

Now, we consider adjusting stepsize to balance $r\|x^k - \tilde{x}^k\|^2$ and $(1/r)(\mu\|AA^T\|\|\lambda^k - \tilde{\lambda}^k\|^2 + \|A^T(\lambda^k - \tilde{\lambda}^k)\|^2)$ and obtain another adaptation strategy (see Algorithm 4).

It is worth noting that too many adjustments of stepsizes by Algorithm 4 might cause the algorithm to diverge in practice, and we therefore restrict these adaptations within a limited number of iterations. If one chooses Algorithm 4, there is no need to carry out Step 5 in Algorithm 1 (or Algorithm 2). These techniques embedded into relaxed-PPA automatically choose a "better" stepsize for the next iteration.

5. Convergence Analysis

In this section, we analyze convergence of our primal-dual relaxed-PPA. The convergence analysis of dual-primal scheme can follow a similar procedure.

Let $u^* = (x^*, \lambda^*)$ be any solution point, setting $u = u^*$ in (32) yields

$$\left(\tilde{u}^k - u^*\right)^T M\left(u^k - \tilde{u}^k\right) \geq \left(\tilde{u}^k - u^*\right)^T F\left(\tilde{u}^k\right). \tag{46}$$

Since $\widetilde{u}^k \in \Omega$, we have $(\widetilde{u}^k - u^*)^T F(u^*) \geq 0$. Consequently, by using the monotonicity of F, the right hand side of (46) is nonnegative, and thus

$$\left(\widetilde{u}^k - u^*\right)^T M\left(u^k - \widetilde{u}^k\right) \geq 0. \tag{47}$$

Now, we are writing the update as

$$u(\alpha) = u^k - \alpha d\left(u^k, \widetilde{u}^k\right) \tag{48}$$

and

$$\begin{aligned} \vartheta(\alpha) &:= \left\|u^k - u^*\right\|_G^2 - \left\|u(\alpha) - u^*\right\|_G^2, \\ q(\alpha) &= 2\alpha\left(u^k - \widetilde{u}^k\right)^T G d\left(u^k, \widetilde{u}^k\right) - \alpha^2 \left\|d\left(u^k, \widetilde{u}^k\right)\right\|_G^2. \end{aligned} \tag{49}$$

Here, $\vartheta(\alpha)$ can be viewed as a progress function. The following lemma shows that $q(\alpha)$ is a lower bound of $\vartheta(\alpha)$.

Lemma 8. *Let $\vartheta(\alpha)$ and $q(\alpha)$ be defined in* (49)*; then one has*

$$\vartheta(\alpha) \geq q(\alpha). \tag{50}$$

Proof. Let u^* be any solution, from the definition of $u(\alpha)$, we have

$$\begin{aligned} \vartheta(\alpha) &= \left\|u^k - u^*\right\|_G^2 - \left\|\left(u^k - u^*\right) - \alpha d\left(u^k, \widetilde{u}^k\right)\right\|_G^2 \\ &= 2\alpha\left(u^k - u^*\right)^T G d\left(u^k, \widetilde{u}^k\right) - \alpha^2 \left\|d\left(u^k, \widetilde{u}^k\right)\right\|_G^2. \end{aligned} \tag{51}$$

Applying (47) to the first term of (51) gives

$$\begin{aligned} \left(u^k - u^*\right)^T G d\left(u^k, \widetilde{u}^k\right) &= \left(u^k - \widetilde{u}^k\right)^T G d\left(u^k, \widetilde{u}^k\right) \\ &\quad + \left(\widetilde{u}^k - u^*\right)^T G d\left(u^k, \widetilde{u}^k\right) \\ &\geq \left(u^k - \widetilde{u}^k\right)^T G d\left(u^k, \widetilde{u}^k\right). \end{aligned} \tag{52}$$

Substituting (52) into (51), we immediately obtain the assertion. □

We note that $q(\alpha)$ is a quadratic function of α and it is natural to maximize $q(\alpha)$ to obtain an "optimal" α:

$$\alpha_k^* = \frac{\left(u^k - \widetilde{u}^k\right)^T G d\left(u^k, \widetilde{u}^k\right)}{\left\|d\left(u^k, \widetilde{u}^k\right)\right\|_G^2}. \tag{53}$$

We now show that the "optimal" α_k^* is bounded above from zero in the following Lemma.

Lemma 9. *Let sequence $\{u^k\}$ be produced by Algorithm 1, α_k^* defined in* (53)*; then, one has*

$$\alpha_k^* \geq \frac{1}{4} > 0. \tag{54}$$

Proof. Using the definition of α_k^* in (53), we have, for all k,

$$\alpha_k^* = \frac{\varphi\left(u^k, \widetilde{u}^k\right)}{\left\|d\left(u^k, \widetilde{u}^k\right)\right\|_G^2} \geq \frac{1}{4}. \tag{55}$$

The inequality follows from condition (#). □

Setting $\alpha = \alpha_k = \alpha_k^* \gamma$ in (50) yields

$$\begin{aligned} \left\|u^{k+1} - u^*\right\|_G^2 &\leq \left\|u^k - u^*\right\|_G^2 \\ &\quad - \gamma(2-\gamma)\alpha_k^*\left(u^k - \widetilde{u}^k\right)^T G d\left(u^k, \widetilde{u}^k\right). \end{aligned} \tag{56}$$

Combining Lemmas 8 and 9, we immediately obtain the following convergence theorem.

Theorem 10. *Let sequence $\{u^k\}$ be produced by Algorithm 1; then one gets*

$$\left\|u^{k+1} - u^*\right\|_G^2 \leq \left\|u^k - u^*\right\|_G^2 - \frac{\gamma(2-\gamma)}{16}\left\|d\left(u^k, \widetilde{u}^k\right)\right\|_G^2. \tag{57}$$

Theorem 11. *Let sequence $\{u^k\}$ be generated by Algorithm 1. Then, $\{u^k\}$ converges to some u^∞ which is a solution of $VI(\Omega, F)$* (9).

Proof. First, for each $u \in \Omega$, we have

$$\left(u - \widetilde{u}^k\right)^T F\left(\widetilde{u}^k\right) \geq \left(u - \widetilde{u}^k\right)^T M\left(u^k - \widetilde{u}^k\right). \tag{58}$$

It follows from (57) that $\{u^k\}$ is a bounded sequence and

$$\lim_{k\to\infty}\left\|u^k - \widetilde{u}^k\right\|_G = 0. \tag{59}$$

Consequently, $\{\widetilde{u}^k\}$ is also bounded. Since $\lim_{k\to\infty}\|u^k - \widetilde{u}^k\|_G = 0$, it follows from (58) that

$$\lim_{k\to\infty}\left(u - \widetilde{u}^k\right)^T F\left(\widetilde{u}^k\right) \geq 0, \quad \forall u \in \Omega. \tag{60}$$

Because $\{\widetilde{u}^k\}$ is bounded, it has at least a cluster point. Let u^∞ be a cluster point of $\{\widetilde{u}^k\}$ and let the subsequence $\{\widetilde{u}^{k_j}\}$ converge to u^∞. It follows that

$$\lim_{j\to\infty}\left(u - \widetilde{u}^{k_j}\right)^T F\left(\widetilde{u}^{k_j}\right) \geq 0, \quad \forall u \in \Omega, \tag{61}$$

and consequently,

$$\left(u - u^\infty\right)^T F\left(u^\infty\right) \geq 0. \quad \forall u \in \Omega. \tag{62}$$

This means that u^∞ is a solution of $\mathrm{VI}(\Omega, F)$. Note that the inequality (57) is true for all solution points of $\mathrm{VI}(\Omega, F)$, and hence, we have

$$\left\|u^{k+1} - u^\infty\right\|_G^2 \leq \left\|u^k - u^\infty\right\|_G^2, \quad \forall k \geq 0. \tag{63}$$

Since $\widetilde{u}^{k_j} \to u^\infty$ $(j \to \infty)$ and $u^k - \widetilde{u}^k \to 0$ $(k \to \infty)$, for any given $\varepsilon > 0$, there exists an integer $l > 0$ such that

$$\left\|\widetilde{u}^{k_l} - u^\infty\right\|_G < \frac{\varepsilon}{2}, \qquad \left\|u^{k_l} - \widetilde{u}^{k_l}\right\|_G < \frac{\varepsilon}{2}. \tag{64}$$

Therefore, for any $k \geq k_l$, it follows from (63) and (64) that

$$\begin{aligned} \left\|u^k - u^\infty\right\|_G &\leq \left\|u^{k_l} - u^\infty\right\|_G \\ &\leq \left\|u^{k_l} - \widetilde{u}^{k_l}\right\|_G + \left\|\widetilde{u}^{k_l} - u^\infty\right\|_G \leq \varepsilon. \end{aligned} \tag{65}$$

This implies that the sequence $\{u^k\}$ converges to u^∞, which is a solution of $\mathrm{VI}(\Omega, F)$ (9). □

Table 1: Comparison of behaviours of four SRPPAs.

A	SPDRPPAG1-I		SDPRPPAG1-I		SPDRPPAG2-I		SDPRPPAG2-I	
n	It.	CPU	It.	CPU	It.	CPU	It.	CPU
500	360	0.13	241	0.13	513	0.19	875	0.26
1500	379	1.32	259	0.89	387	1.72	594	2.04
2500	501	4.39	343	3.01	429	4.93	648	5.72
3500	399	6.65	325	5.57	663	14.44	1089	18.02
4500	389	12.13	457	14.47	493	20.61	991	31.26
5500	515	24.21	462	21.97	497	30.88	813	38.43
6500	449	25.01	423	23.41	456	33.42	932	51.62
7500	520	45.72	473	41.79	724	63.06	886	77.91

6. Applications and Preliminary Numerical Experiments

The general self-adaptive relaxed-PPA (SRPPA) offers a flexible framework for solving many interesting problems. We illustrate our algorithm with different applications: basis pursuit problem, nearest correlation matrix problem. In this section, we describe the results of experiments whose goal is to demonstrate the efficiency of general relaxed-PPA (RPPA) and its self-adaptive version. To that end, we compare RPPA with certain state-of-the-art algorithms on different problems. Our experiments focus on efficiency and speed of convergence and evaluate the methods in terms of their number of iterations and computational times.

All the codes were written by Matlab R2009b version, and all the numerical experiments were performed on a Lenovo desktop computer with Intel (R) Core (TM) i5 CPU with 3.2 GHz and 3.5 GB RAM.

6.1. Basis Pursuit Problem. Basis pursuit (BP) finds signal representations in overcomplete dictionaries by equality-constrained l_1 minimization problem. Formally, one solves the problem

$$\min \left\{ \|x\|_1 Ax = b,\ x \in \Re^n \right\}. \tag{66}$$

And here, $\|\cdot\|_1$ denotes the l_1 norm defined as $\|x\|_1 := \sum_{i=1}^{n} |x_i|$. BP is a fundamental problem in image processing and modern statistical signal processing, particularly the theory of compressed sensing; see, for example, [1–4] for intensive study. We now discuss our approach to BP problem of over-complete representations. Our experiments in this subsection use synthetic data which were mainly designed to illustrate the nice performance of our RPPA. The synthetic problem that we test here is similar to the one employed in [32]. We generate the data as follows: matrix A is a random $m \times n$ matrix, with Gaussian i.i.d. entries of zero mean and variance 1, with $m = n/2$. $x_{\text{original}} \in R^n$ is the original sparse signal, constructed with $m/5$ nonzero values, randomly from standard normal distribution. We use x_{original} to generate the measurements as $b = Ax_{\text{original}}$. It is desirable to use test problems that have a precisely known solution. In fact, when x_{original} is very sparse, it is the solution to the minimization problem (66). Hence, in our synthetic problem, x_{original} is exactly the solution.

In our first experiment, we compared general RPPA using two different G's mentioned in Section 4.1. For BP problem, we use condition (#) and Algorithm 3. Since A constructed here is a general random matrix, and when A is large scale, $\|A^T A\|$ might be obtained costly. A simple stopping criterion

$$\text{err}. = \left\| x^k - x_{\text{original}} \right\| \le \text{Tol} \tag{67}$$

was used in this experiment, and the stopping tolerance Tol was set to 10^{-10}. In all the tests, initial stepsizes were set as $s = 10$, $r = 1$, the primal variable x^0 was initialized as zeros$(n, 1)$, and the dual λ^0 was ones$(m, 1)$ in Matlab. Table 1 summarizes the performance of general SRPPA. Here, SPDRPPAGi-I(SPDRPPAGi-II) denotes self-adaptive primal-dual RPPA with Algorithm 3 (Algorithm 4), $G = H$, if $i = 1$, and $G = MH^{-1}M^T$, if $i = 2$. DPRPPA (DPRPPA) denotes dual-primal RPPA version.

Basically, SRPPAs converge very quickly and achieved tight error 10^{-10} in a few hundred iterations. For this experiment, one can see that SDPRPPAG1-I is fastest in all cases. Both SDPRPPAG1-I and SPDRPPAG1-I are Gaussian type methods, with $G = H$, and they exhibit very similar performance. SDPRPPAG2-I and SPDRPPAG2-I with $G = MH^{-1}M^T$ are Gaussian back substitute form methods and perform a little slower than Gaussian type methods. We also plot a figure to graphically illustrate the performance of four SRPPAs. Figure 1 shows the results from the test with $n = 1000$ and $n = 6000$, depicting error versus CPU time. Quality-wise, SPDRPPAG1-I was on par with SDPRPPAG1-I.

In the second experiment, we compare the performance of SPDRPPAG1-I with TFOCS (source code can be found at http://cvxr.com/tfocs/) [32], ADMM (source code can be found at http://www.stanford.edu/~boyd/papers/admm/) [33], and PPA-CM. To make the comparison independent of the stopping criterion for each algorithm, we first run TFOCS to get its solution x_{TFOCS} and set a benchmark error

$$\text{benchmark err}. = \left\| x_{\text{TFOCS}} - x_{\text{original}} \right\|_2 \tag{68}$$

and then run other algorithms until they obtain smaller errors than this benchmark. TFOCS was stopped upon

$$\frac{\left\| x^{k+1} - x^k \right\|}{\max\left\{1, \left\| x^{k+1} \right\|\right\}} \le \text{Tol}. \tag{69}$$

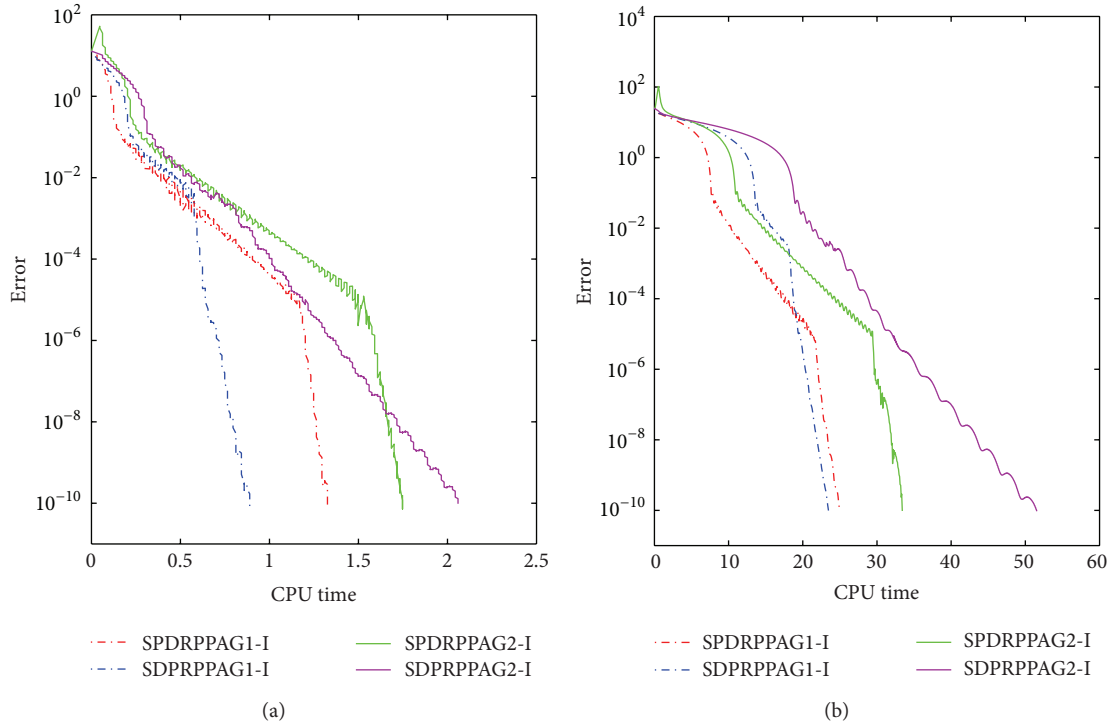

Figure 1: Comparing SRPPAs applied to BP problem with $n = 1000$ (a) and $n = 6000$ (b). The horizontal axis gives the CPU time; the vertical axis gives the error between the solution and the original.

Table 2: Performance of different iterative methods.

A	TFOCS		ADMM		PPA-CM		SPDRPPAG1-I	
n	It./CPU	Err.	It./CPU	Err.	It./CPU	Err.	It./CPU	Err.
512	1681/4.16	$7.5e-10$	492/0.67	$5.0e-10$	1027/1.41	$6.2e-10$	391/0.10	$4.0e-10$
1024	406/1.68	$3.7e-10$	274/0.53	$3.0e-10$	848/0.76	$2.3e-10$	214/0.28	$3.0e-10$
2048	507/4.01	$4.4e-9$	607/3.93	$2.9e-9$	804/3.65	$4.0e-9$	239/1.49	$3.9e-9$
4096	933/19.50	$2.6e-9$	1070/27.47	$2.1e-9$	845/15.09	$2.3e-9$	422/9.84	$2.3e-9$
4200	461/9.91	$2.1e-9$	916/25.48	$1.8e-9$	868/16.27	$2.1e-9$	391/9.44	$1.5e-9$
4300	451/9.74	$2.2e-9$	464/20.37	$1.6e-9$	884/16.91	$2.1e-9$	429/10.54	$2.1e-9$
4600	505/12.37	$1.36e-9$	2155/56.45	$1.2e-9$	863/18.95	$1.2e-9$	425/11.89	$1.1e-9$
4700	801/20.15	$1.3e-11$	1517/45.37	$8.2e-12$	1102/34.27	$1.2e-11$	425/14.20	$1.1e-11$
5500	407/13.64	$2.4e-6$	—/—	—	546/20.13	$1.9e-6$	308/12.40	$2.0e-6$
6500	1257/56.11	$1.5e-5$	—/—	—	508/26.48	$1.4e-5$	308/17.22	$1.3e-5$
7500	801/81.42	$1.1e-12$	—/—	—	1313/813.52	$1.1e-12$	522/69.52	$1.1e-12$
8500	842/107.18	$2.3e-7$	—/—	—	724/100.36	$2.2e-7$	542/83.07	$2.2e-7$

Since we found that Tol $= 10^{-12}$ is small enough to guarantee very high accuracy, we set Tol $= 10^{-12}$ in TFOCS. The parameters of TFOCS and ADMM were taken with their defaults. To guarantee the convergence, fixed stepsizes r, s were set to $s = 100$, $r = 1.01 * \|AA^T\|/s$ for PPA-CM. In SPDRPPAG1-I, we also choose the same convergence condition (#) and initial step size $s = 10$, $r = 1$ as the previous experiment. We varied the size of A from $n = 512$ ($m = n/2$) to $n = 8500$. The results of this experiment are summarized in Table 2. There, we report the run time in seconds, the number of iterations, and the error of the recovery solution. In Table 2, "—" means "out of memory."

We observe from Table 2 that four algorithms reach high accuracy around 10^{-9}. SPDRPPAG1-I is about two times faster than the first-order method implemented in the TFOCS package, and moreover, it usually outperforms TFOCS in terms of iterations. For medium size problems, SPDRPPAG1-I is clearly faster than ADMM. Even for small size problems, SPDRPPAG1-I shows its superior performance. The main reason lies in that ADMM computed $(I + AA^T)^{-1}$ to solve its subproblem exactly which would take expensive computational cost. Not surprisingly, the general SPDRPPAG1-I performs better than the primary PPA-CM. Here, the total iterations of SPDRPPAG1-I are less than 50%

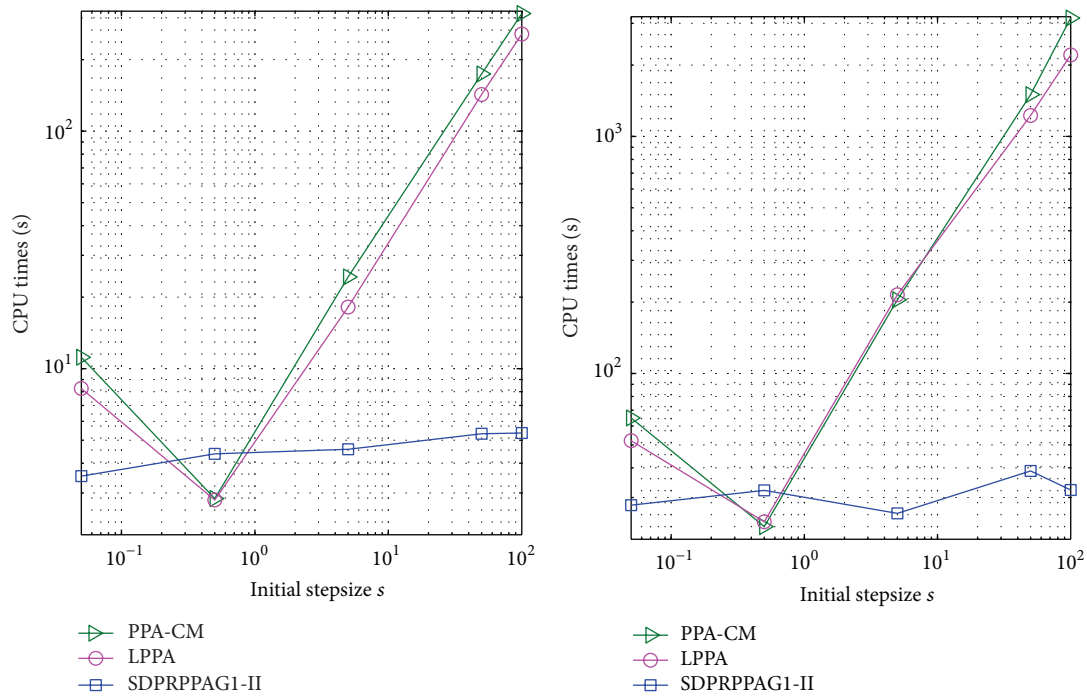

Figure 2: CPU times as a function of the initial stepsize s for PPA-CM, LPPA, and SDPRPPAG1-II. The plot on the left is for $n = 500$, while the plot on the right is for $n = 1000$.

of PPA-CM. As we have mentioned, "optimal" update stepsize α_k and more flexible condition for convergence may provide SPDRPPAG1-I improved performance. SPDRPPAG1-I is faster than PPA-CM in terms of CPU times. However, the superiority of CPU time is not so significant as iteration number. For the cases $n = 4300$, it is just about 62% of PPA-CM. This is not particularly surprising; compared to PPA-CM, SPDRPPAG1-I has to take extra computation for convergence condition and "optimal" α_k in each iteration.

6.2. Nearest Correlation Matrix Problem. The nearest correlation matrix problem is solving the problem

$$\min \left\{ \frac{1}{2} \|X - C\|_F^2 \mid \operatorname{diag}(X) = e,\ X \in S_+^n \right\}, \qquad (70)$$

where $e \in \Re^n$ is the vector whose entries are all 1s, S_+^n denotes the cone of positive definite symmetric matrices, $\operatorname{diag}(X)$ is the vector of diagonal elements of X, and $\|\cdot\|_F$ denotes the matrix Fröbenius norm $\|X\|_F = \operatorname{trace}(X^T X)^{1/2}$.

Here, we apply PPA-CM, LPPA, and SDPRPPA1-II for solving (70). The standard Matlab Mex interface mexsvd is used to conduct the eigenvalue decomposition. We constructed test data sets and stopping criterion like those of [24]. As mentioned in the prequel, we expect our SRPPA to produce robust performance. To assess the effectiveness of the adaptive strategies proposed in Section 6, we now move on to the description of experiments that focus on the consequences of the initial stepsizes. For investigating, we used dimensions $n \in \{500, 1000\}$ and varied s from 0.05 to 100, and initial points were set to 0 in all cases. Note that $A = I$; we fixed $r = 1.01/s$ for PPA-CM, $r = 0.65/s$ for LPPA and chose $r = 0.65/s$ as initial start for SDPRPPAG1-II. Since the experiments with other values of n give qualitatively similar results, we therefore do not plot those results to avoid clutter in the figures. The respective numerical results are plotted in Figure 2.

It is clear that, for PPA-CM and LPPA, the convergence performance was a result of the stepsize selection. They are both fairly sensitive to initial stepsize s (or r). The results confirm that, with inappropriate stepsizes, both PPA-CM and LPPA become significantly slow. SDPRPPAG1-II yields significantly robust performance with adaptive strategy. And it is independent of the initial stepsizes and illustrates its superior performance. Furthermore, SDPRPPAG1-II yields competitive results even when PPA-CM and LPPA chose the "good" initial stepsize. This underlies the importance of adaptive strategy in producing good performance. Of course, care should be taken. For instance, the cost of computing optimal stepsize α_k here is negligible, compared to the computation of SVD; when they are more costly, general LPPA will be expected to perform slower than PPA-CM.

7. Conclusions

In this paper, we proposed an efficient general self-adaptive relaxed-PPA method for linearly constrained convex programming and provided theoretical convergence analysis for

this method. The stepsizes choice condition is flexible and simple. Self-adaptive strategies are provided to make our method more efficient and robust. Experiments of the method in comparison to other state-of-art methods are provided to confirm the efficiency of the proposed method. Our numerical results suggest that SRPPA is effective and simple to implement. There are a few directions for further research, but we list here only two. The first is the question of whether we may modify the algorithm to work with more general constrained convex problems. Second, we aim to provide various relaxations of the subproblem for the practical purpose.

Acknowledgments

The author is grateful to Caihua Chen, Wenxing Zhang, and Xiangfeng Wang for interesting discussions on nearest correlation matrix problem. Xiaoling Fu was supported by the NSFC Grant 70901018.

References

[1] A. M. Bruckstein, D. L. Donoho, and M. Elad, "From sparse solutions of systems of equations to sparse modeling of signals and images," *SIAM Review*, vol. 51, no. 1, pp. 34–81, 2009.

[2] E. J. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 489–509, 2006.

[3] E. J. Candès and T. Tao, "Near-optimal signal recovery from random projections: universal encoding strategies?" *IEEE Transactions on Information Theory*, vol. 52, no. 12, pp. 5406–5425, 2006.

[4] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Review*, vol. 43, no. 1, pp. 129–159, 2001.

[5] R. Borsdorf and N. J. Higham, "A preconditioned Newton algorithm for the nearest correlation matrix," *IMA Journal of Numerical Analysis*, vol. 30, no. 1, pp. 94–107, 2010.

[6] S. Boyd and L. Xiao, "Least-squares covariance matrix adjustment," *SIAM Journal on Matrix Analysis and Applications*, vol. 27, no. 2, pp. 532–546, 2005.

[7] N. J. Higham, "Computing the nearest correlation matrix—a problem from finance," *IMA Journal of Numerical Analysis*, vol. 22, no. 3, pp. 329–343, 2002.

[8] J.-F. Cai, E. J. Candès, and Z. Shen, "A singular value thresholding algorithm for matrix completion," *SIAM Journal on Optimization*, vol. 20, no. 4, pp. 1956–1982, 2010.

[9] E. J. Candès and B. Recht, "Exact matrix completion via convex optimization," *Foundations of Computational Mathematics*, vol. 9, no. 6, pp. 717–772, 2009.

[10] E. J. Candès and T. Tao, "The power of convex relaxation: near-optimal matrix completion," *IEEE Transactions on Information Theory*, vol. 56, no. 5, pp. 2053–2080, 2010.

[11] B. Martinet, "Régularisation d'inéquations variationnelles par approximations successives," *Revue Franqaise dInformatique et de Recherche Operationelle*, vol. 4, pp. 154–158, 1970.

[12] A. Bnouhachem and M. A. Noor, "Inexact proximal point method for general variational inequalities," *Journal of Mathematical Analysis and Applications*, vol. 324, no. 2, pp. 1195–1212, 2006.

[13] A. Bnouhachem and M. A. Noor, "An interior proximal point algorithm for nonlinear complementarity problems," *Nonlinear Analysis: Hybrid Systems*, vol. 4, no. 3, pp. 371–380, 2010.

[14] J. V. Burke and M. Qian, "A variable metric proximal point algorithm for monotone operators," *SIAM Journal on Control and Optimization*, vol. 37, no. 2, pp. 353–375, 1999.

[15] J. Eckstein, "Approximate iterations in Bregman-function-based proximal algorithms," *Mathematical Programming*, vol. 83, no. 1, pp. 113–123, 1998.

[16] J. Eckstein and P. J. S. Silva, "Proximal methods for nonlinear programming: double regularization and inexact subproblems," *Computational Optimization and Applications*, vol. 46, no. 2, pp. 279–304, 2010.

[17] O. Güler, "On the convergence of the proximal point algorithm for convex minimization," *SIAM Journal on Control and Optimization*, vol. 29, no. 2, pp. 403–419, 1991.

[18] M. R. Hestenes, "Multiplier and gradient methods," *Journal of Optimization Theory and Applications*, vol. 4, pp. 303–320, 1969.

[19] R. T. Rockafellar, "Monotone operators and the proximal point algorithm," *SIAM Journal on Control and Optimization*, vol. 14, no. 5, pp. 877–898, 1976.

[20] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends in Optimization*, vol. 1, no. 3, pp. 1–108, 2013.

[21] R. T. Rockafellar, "Augmented Lagrangians and applications of the proximal point algorithm in convex programming," *Mathematics of Operations Research*, vol. 1, no. 2, pp. 97–116, 1976.

[22] G. M. Korpelevič, "An extragradient method for finding saddle points and for other problems," *Èkonomika i Matematicheskie Metody*, vol. 12, no. 4, pp. 747–756, 1976.

[23] B. S. He and X. M. Yuan, "A contraction method with implementable proximal regularization for linearly constrained convex programming," *Optimization Online*, pp. 1–14, 2010.

[24] Y. F. You, X. L. Fu, and B. S. He, "Lagrangian PPA-based contractionmethods for linearly constrained convex optimization," *Pacific Journal of Optimization*. In press.

[25] B. S. He, X. L. Fu, and Z. K. Jiang, "Proximal-point algorithm using a linear proximal term," *Journal of Optimization Theory and Applications*, vol. 141, no. 2, pp. 299–319, 2009.

[26] R. T. Rockafellar, *Convex Analysis*, Princeton Mathematical Series, no. 28, Princeton University Press, Princeton, NJ, USA, 1970.

[27] F. Facchinei and J. S. Pang, *Finite-Dimensional Variational Inequalities and Complementarity Problems*, Springer, New York, NY, USA, 2003.

[28] B. He, "A class of projection and contraction methods for monotone variational inequalities," *Applied Mathematics and Optimization*, vol. 35, no. 1, pp. 69–76, 1997.

[29] B. S. He and L. Z. Liao, "Improvements of some projection methods for monotone nonlinear variational inequalities," *Journal of Optimization Theory and Applications*, vol. 112, no. 1, pp. 111–128, 2002.

[30] B. He, X. Yuan, and J. J. Z. Zhang, "Comparison of two kinds of prediction-correction methods for monotone variational inequalities," *Computational Optimization and Applications*, vol. 27, no. 3, pp. 247–267, 2004.

[31] X. Fu and B. He, "Self-adaptive projection-based prediction-correction method for constrained variational inequalities," *Frontiers of Mathematics in China*, vol. 5, no. 1, pp. 3–21, 2010.

[32] S. R. Becker, E. J. Candès, and M. C. Grant, "Templates for convex cone problems with applications to sparse signal

recovery," *Mathematical Programming Computation*, vol. 3, no. 3, pp. 165–218, 2011.

[33] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2010.