Convergent prediction-correction-based ADMM for multi-block separable convex programming *

Xiaokai Chang[a,b†], Sanyang Liu[a], Pengjun Zhao[c] and Xu Li[b]

[a]School of Mathematics and Statistics, Xidian University, Xi'an 710071, P. R. China.

[b]College of Science, Lanzhou University of Technology, Lanzhou, Gansu 730050, P. R. China.

[c]School of Mathematics and Computer Application, Shanglue University, Shanglue, 726000, P. R. China.

December 16, 2017

Abstract

The direct extension of the classic alternating direction method with multipliers (ADMMe) to the multi-block separable convex optimization problem is not necessarily convergent, though it often performs very well in practice. In order to preserve the numerical advantages of ADMMe and obtain convergence, many modified ADMM were proposed by correcting the output of ADMMe or employing proximal terms to solve inexactly the subproblems in ADMMe. In this paper, we present an efficient Prediction-Correction-based ADMM (PCB-ADMM) to solve the multi-block separable convex minimization model. The prediction step takes a special block coordinate descent (BCD) cycle to update the variable blocks, then the correction step corrects the output slightly by computing a convex combination of two points from the prediction step and previous iteration. The convergence property is obtained by using the variational inequality. The numerical experiments illustrate effectiveness of the proposed PCB-ADMM to solve the quadratic semidefinite programming and image decomposition.

Key words. alternating direction method of multipliers, prediction-correction, variational inequality, convergence analysis, quadratic semidefinite programming, image decomposition.

1 Introduction.

The convex minimization problem with linear constraints and a separable objective function has numerous applications in several areas, such as the dual of the (quadratic) semidefinite programming (SDP) with or without nonnegative constraints [1, 2, 3], the robust principal component analysis model with noisy and incomplete data [4, 5], image decomposition [6, 7, 8], and so on. In this paper, we consider the following separable convex minimization model with n block variables and

*This work was supported in part by the National Natural Science Foundation of China under Grant 61373174 and 11501272.

†Corresponding author, e-mail: xkchang@lut.cn.

the objective being the sum of $n(n \geq 3)$ separable convex functions:

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} \theta_i(x_i) \\ s.t. \quad & \sum_{i=1}^{n} \mathcal{A}_i(x_i) = c, \quad x_i \in \mathcal{X}_i, i = 1, \cdots, n. \end{aligned} \tag{1}$$

Generally, $\theta_i : \mathcal{X}_i \to \mathbb{R}$ $(i = 1, \cdots, n)$ in the model (1) are closed proper convex functions (not necessarily smooth), and $\mathcal{X}_i$ $(i = 1, \cdots, n)$ are finite dimensional real Euclidean spaces or closed convex sets. $\mathcal{A}_i : \mathcal{X}_i \to \mathbb{R}^n$ $(i = 1, \cdots, n)$ are given linear operators and $c \in \mathbb{R}^n$ is a given vector. Throughout this paper, the solution set of the problem (1) is assumed to be nonempty.

The augmented lagrangian function of the problem (1) has the following form:

$$\mathcal{L}_\sigma(x_1, \cdots, x_n, \lambda) = \sum_{i=1}^{n} \theta_i(x_i) - \langle \lambda, \sum_{i=1}^{n} \mathcal{A}_i(x_i) - c \rangle + \frac{\sigma}{2} \| \sum_{i=1}^{n} \mathcal{A}_i(x_i) - c \|^2, \tag{2}$$

where $\lambda \in \mathbb{R}^n$, $\sigma > 0$ is a penalty parameter, $\langle \cdot, \cdot \rangle$ represents the standard inner product and $\| \cdot \|$ represents the l_2-norm throughout.

For the denotational convenience, we define

$$x_{(h:k)} = (x_h, x_{h+1}, \cdots, x_k), \tag{3}$$

for given positive integers h and $k(k \geq h)$, with the convention that $x_{(h:h)} = x_h$.

Given a chosen initial point, the direct extension of the classic ADMM (ADMMe) to the n-block convex optimization problem (1) consists of the iterations:

$$\left.\begin{aligned} x_1^{k+1} &:= \arg\min_{x_1 \in \mathcal{X}_1} \mathcal{L}_\sigma(x_1, x_{(2:n)}^k, \lambda^k), \\ &\vdots \\ x_i^{k+1} &:= \arg\min_{x_i \in \mathcal{X}_i} \mathcal{L}_\sigma(x_{(1:i-1)}^{k+1}, x_i, x_{(i+1:n)}^k, \lambda^k), \\ &\vdots \\ x_n^{k+1} &:= \arg\min_{x_n \in \mathcal{X}_n} \mathcal{L}_\sigma(x_{(1:n-1)}^{k+1}, x_n, \lambda^k), \\ \lambda^{k+1} &:= \lambda^k - \tau\sigma \left(\sum_{i=1}^{n} \mathcal{A}_i(x_i^{k+1}) - c \right), \end{aligned}\right\} \tag{4}$$

where $\tau > 0$, e.g., $\tau \in (0, \frac{1+\sqrt{5}}{2})$, is a positive constant that controls the step-length in (4).

It was showed by a counter example that the convergence of (4) is failure in [9], though ADMMe has many advantages, such as the extreme simplicity and efficiency. In order to guarantee convergence and preserve the numerical advantages of ADMMe, many ADMM's variants were developed based on different methods, see [1, 2]. One is to add a simple correction step, for example, a convergent alternating direction method with a Gaussian back substitution (ADM-G) [7, 8], in which the correction step was implemented after ADMMe in each iteration. Another is to employ a simple proximal term to solve inexactly each subproblem, which has been investigated by many researchers, see [1, 10, 11]. In addition, the combination of the above two ways was used, and the algorithm twisted from generalized ADMM was proposed in [12].

Due to the lack of ADMMe's convergence, ADMMe with sufficiently small σ was discussed for solving some special problems with strongly convex block [13], and other methods with different

conditions were introduced [14]-[22]. However, the restriction of σ on a small interval may hinder its effective adjustment according the progress of algorithm, then it will reduce the efficiency of ADMMe. Wang et.al. reformulated the multiblock separable problem as an equivalent two-block problem in [23], and then solved it using the classic two-block ADMM. Furthermore, the convergence result and improved $\mathcal{O}(1/\epsilon)$ iteration complexity result are obtained by the classic 2-block ADMM, though the numerical result is not as effective as that of the directly extended ADMM.

Here, we focus on a special separable convex programming problem. For the objective functions θ_i $(i = 2, \cdots, n-1)$ and the sets $\mathcal{X}_i$ $(i = 2, \cdots, n-1)$ in the model (1), we make the following assumption:

Assumption 1 *(1). For $i = 2, \cdots, n-1$,*

$$\theta_i(x_i) = \frac{1}{2}\langle x_i, \psi_i(x_i)\rangle + \langle b_i, x_i\rangle, \tag{5}$$

where ψ_i is given self-adjoint positive semidefinite linear operator and $b_i \in \mathcal{X}_i$ is constant;

(2). For $i = 2, \cdots, n-1$, $\mathcal{X}_i$ is a finite dimensional real Euclidean space;

(3). The linear operators $\mathcal{A}_i$ $(i = 1, \cdots, n)$ satisfy that, $\mathcal{A}_i^*\mathcal{A}_i$ is invertible. Namely, $\mathcal{A}_i$ is surjective.

Under Assumption 1, the objective functions $\theta_i(i = 2, \cdots, n-1)$ are quadratic convex or linear (if $\psi_i = 0$), and $\partial\theta_i(x_i) = \psi_i(x_i) + b_i$. Moreover, we should mention that, generally, since that $\mathcal{A}_i$ is assumed being surjective, all the subproblems are well-defined and admit unique solutions. The well-definedness of subproblems is very essential for ADMM-type algorithm. On this part, one may refer to a counterexample by Chen et al. [24]. For the special case of the problem (1) with $n = 3$ and θ_2 being linear, Sun et.al., proposed a convergent 3-block ADMM (ADMMc3b) [1] by taking one special block coordinate descent (BCD) cycle with the order $1 \to 2 \to 3 \to 2$. The convergence of ADMMc3b was obtained by showing the equivalence of it to the classic 2-block semi-proximal ADMM for solving its 2-block reformulation. Based on Schur complement, a convergent semi-proximal ADMM (SCB-ADMM) was introduced for multi-block separable convex programming in [10], its convergence was obtained by using the same strategy as in [1].

The dual of the convex quadratic SDP with (or without) nonnegative constraints is a 4-block (or 3-block) separable convex problem, Chang et al. presented a modified ADMM (MADMM) in [2] for solving this dual. At each iteration, the proposed methods require no correction steps and no computation of the subproblem with the primal variable X, to generate new iterates. The convergence of MADMM can be obtained when some moderate conditions are satisfied on the penalty parameter σ.

More recently, He et.al., presented a class of ADMM-based algorithms [8] for multi-block separable convex programming (1), which suggests correcting the output of ADMMe slightly by a simple correction step. Furthermore, the prediction-correction framework of ADMM-based algorithms was proposed in [8]. In order to ensure the convergence, a unified and easily checkable condition (Convergence Condition in Section 3.1 of [8]) was given. Based on this framework, many efficient methods can be designed by proposing different prediction and correction steps.

Inspired by the aforementioned work on ADMM and its variants, in this paper we propose a prediction-correction-based ADMM (PCB-ADMM) for solving the problem (1) under Assumption 1. The prediction step takes a special BCD cycle with the order $1 \to 2 \to \cdots \to n \to n-1 \to \cdots \to 2$, similar as SCB-ADMM in [10]. The correction step only need to compute a simple

convex combination of two iteration points from the prediction step and previous iteration. The convergence is shown by using the variational inequality. Especially, we obtain the convergence of PCB-ADMM with unit step-length ($\alpha = 1$) in the correction step under a moderate condition. In this case, the correction step is unnecessary, and Convergence Condition defined in [8] can not be guaranteed.

The rest of this paper is organized as follows. We introduce our PCB-ADMM in Section 2. In Section 3, the convergence of the proposed PCB-ADMM is analysed by using the variational inequality. Section 4 is devoted to the implementation and numerical experiments, for solving the CQSDP problems generated randomly with nonnegative constraints and low patch-rank decomposition. We conclude our paper in the final section.

2 Prediction-Correction-based ADMM

Firstly, we give some notations in this section. Then we will introduce our prediction-correction-based ADMM for solving the multi-block separable convex programming (1).

2.1 Notations

We define the following notations that will be used later:

$$u = x_{(1:n)}^T, \quad w = (x_{(1:n)}, \lambda)^T, \quad v = (x_{(2:n)}, \lambda)^T, \quad z = (x_{(3:n)}, \lambda)^T, \tag{6}$$

$$\theta(u) = \sum_{i=1}^{n} \theta_i(x_i), \quad \widehat{\mathcal{A}} = (\mathcal{A}_1, \mathcal{A}_2, \cdots, \mathcal{A}_n), \tag{7}$$

$$\mathcal{F}(w) = \begin{pmatrix} -\mathcal{A}_1^*(\lambda) \\ \vdots \\ -\mathcal{A}_n^*(\lambda) \\ \sum_{i=1}^n \mathcal{A}_i(x_i) - c \end{pmatrix} \quad \text{and} \quad \mathcal{W} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_n \times \mathbb{R}^n. \tag{8}$$

By making use of the notations above, the Lagrangian function of problem (1) can be defined by

$$L(u, \lambda) = \theta(u) - \langle \lambda, \widehat{\mathcal{A}}(u) - c \rangle, \tag{9}$$

for any $(u, \lambda) \in \mathcal{W}$. Then solving (1) is equivalent to finding a saddle point of $L(u, \lambda)$. By the convex analysis [25, Theorem 28.3], $w^* = (u^*, \lambda^*) \in \mathcal{W}$ is a saddle point of $L(u, \lambda)$ if and only if the following variational inequality is satisfied:

$$\theta(u) - \theta(u^*) + \langle w - w^*, \mathcal{F}(w^*) \rangle \geq 0, \quad \forall w \in \mathcal{W}. \tag{10}$$

We denote by $\mathcal{W}^*$ the solution set of the variational inequality (10), and it is nonempty because of the nonemptyness of the solution set of (1).

Moreover, we set

$$v^k = (x_{(2:n)}^k, \lambda^k)^T, \quad v^* = (x_{(2:n)}^*, \lambda^*)^T, \quad \widetilde{v}^k = (\widetilde{x}_{(2:n)}^k, \widetilde{\lambda}^k)^T, \tag{11}$$

$$\mathcal{V} = \mathcal{X}_2 \times \cdots \times \mathcal{X}_n \times \mathbb{R}^n, \quad \mathcal{V}^* = \{(x_{(2:n)}^*, \lambda^*) | (x_{(1:n)}^*, \lambda^*) \in \mathcal{W}^*\}. \tag{12}$$

Similarly, we will use u^k, u^*, $\widetilde{u}^k$, w^k, w^* and $\widetilde{w}^k$, where w^* is a saddle-point of the Lagrangian function associated to problem (1), and that u^* is a solution of problem (1).

2.2 Prediction-Correction-based ADMM

We introduce our prediction-correction-based ADMM for solving (1) and give some properties.

Algorithm 1 (Prediction-Correction-based ADMM for solving (1).)

Step 0. Let $\sigma > 0$ be a given parameter. Choose $v^0 \in \mathcal{V}$. Set $k = 0$.

Step 1. (Prediction) Compute subproblems using the BCD cycle with the order $1 \to 2 \to \cdots \to n \to n-1 \to \cdots \to 2$, and update the Lagrangian multiplier,

$$\begin{cases} \widetilde{x}_1^k = \arg\min_{x_1 \in \mathcal{X}_1} \mathcal{L}_\sigma(x_1, x_{(2:n)}^k, \lambda^k), \\ \widehat{x}_2^k = \arg\min_{x_2 \in \mathcal{X}_2} \mathcal{L}_\sigma(\widetilde{x}_1^k, x_2, x_{(3:n)}^k, \lambda^k), \\ \widehat{x}_j^k = \arg\min_{x_j \in \mathcal{X}_j} \mathcal{L}_\sigma(\widetilde{x}_1^k, \widehat{x}_{(2:j-1)}^k, x_j, x_{(j+1:n)}^k, \lambda^k), \quad \text{for} \quad j = 3, \cdots, n-1, \\ \widetilde{x}_n^k = \arg\min_{x_n \in \mathcal{X}_n} \mathcal{L}_\sigma(\widetilde{x}_1^k, \widehat{x}_{(2:n-1)}^k, x_n, \lambda^k), \\ \widetilde{x}_j^k = \arg\min_{x_j \in \mathcal{X}_j} \mathcal{L}_\sigma(\widetilde{x}_1^k, \widehat{x}_{(2:j-1)}^k, x_j, \widetilde{x}_{(j+1:n)}^k, \lambda^k), \quad \text{for} \quad j = n-1, \cdots, 3, \\ \widetilde{x}_2^k = \arg\min_{x_2 \in \mathcal{X}_2} \mathcal{L}_\sigma(\widetilde{x}_1^k, x_2, \widetilde{x}_{(3:n)}^k, \lambda^k), \\ \widehat{\lambda}^k = \lambda^k - \tau\sigma\left(\sum_{i=1}^n \mathcal{A}_i(\widetilde{x}_i^k) - c\right) \end{cases} \tag{13}$$

Step 2. (Correction) Correct the output from (13) by

$$\begin{cases} x_1^{k+1} = \widetilde{x}_1^k \\ x_{(2:n)}^{k+1} = x_{(2:n)}^k - \alpha(x_{(2:n)}^k - \widetilde{x}_{(2:n)}^k) \\ \lambda^{k+1} = \lambda^k - \alpha(\lambda^k - \widehat{\lambda}^k) \end{cases} \tag{14}$$

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

Step 3. Replace k by $k+1$, and return to Step 1.

Remark 1 *From Step 1, the subproblems with the variable x_i $(i = 2, \cdots, n-1)$ are computed twice. But the correction step is just to implement a simple convex combination for $\alpha \in (0, 1)$. If $\alpha = 1$ in the correction step, the correction is unnecessary to implement. In order to obtain the convergence, we set $\tau = 1$ in Algorithm 1.*

Remark 2 *The convergence of PCB-ADMM with $\alpha \in (0, 1)$ can be obtained directly by checking the Convergence Condition defined in [8, Section 3.1]. However, this Convergence Condition can not be established for PCB-ADMM with $\alpha = 1$. Thus, when the linear operator $\mathcal{A}_2$ is invertible we obtain its convergence by using the variational inequality, see Section 3.*

The optimality conditions of the iterations in the prediction step can be summarized as the following

$$\begin{cases} \widetilde{x}_1^k \in \mathcal{X}_1, \quad \forall x_1 \in \mathcal{X}_1, \\ \quad \theta_1(x_1) - \theta_1(\widetilde{x}_1^k) + \langle x_1 - \widetilde{x}_1^k, -\mathcal{A}_1^*[\lambda^k - \sigma(\mathcal{A}_1(\widetilde{x}_1^k) + \sum_{i=2}^n \mathcal{A}_i(x_i^k) - c)]\rangle \geq 0, \\ \widehat{x}_j^k \in \mathcal{X}_j, \quad \forall x_j \in \mathcal{X}_j, \ j = 2, \cdots, n-1, \\ \quad \theta_j(x_j) - \theta_j(\widehat{x}_j^k) + \left\langle x_j - \widehat{x}_j^k, -\mathcal{A}_j^*[\lambda^k - \sigma(\mathcal{A}_1(\widetilde{x}_1^k) + \sum_{i=2}^j \mathcal{A}_i(\widehat{x}_i^k) + \sum_{i=j+1}^n \mathcal{A}_i(x_i^k) - c)]\right\rangle \geq 0, \\ \widetilde{x}_n^k \in \mathcal{X}_n, \quad \forall x_n \in \mathcal{X}_n, \\ \quad \theta_n(x_n) - \theta_n(\widetilde{x}_n^k) + \left\langle x_n - \widetilde{x}_n^k, -\mathcal{A}_n^*[\lambda^k - \sigma(\mathcal{A}_1(\widetilde{x}_1^k) + \sum_{i=2}^{n-1} \mathcal{A}_i(\widehat{x}_i^k) + \mathcal{A}_n(\widetilde{x}_n^k) - c)]\right\rangle \geq 0, \\ \widetilde{x}_j^k \in \mathcal{X}_j, \quad \forall x_j \in \mathcal{X}_j, \quad j = n-1, \cdots, 3, \\ \quad \theta_j(x_j) - \theta_j(\widetilde{x}_j^k) + \left\langle x_j - \widetilde{x}_j^k, -\mathcal{A}_j^*[\lambda^k - \sigma(\mathcal{A}_1(\widetilde{x}_1^k) + \sum_{i=2}^{j-1} \mathcal{A}_i(\widehat{x}_i^k) + \sum_{i=j}^n \mathcal{A}_i(\widetilde{x}_i^k) - c)]\right\rangle \geq 0, \\ \widetilde{x}_2^k \in \mathcal{X}_2, \quad \forall x_2 \in \mathcal{X}_2, \\ \quad \theta_2(x_2) - \theta_2(\widetilde{x}_2^k) + \langle x_2 - \widetilde{x}_2^k, -\mathcal{A}_2^*[\lambda^k - \sigma(\sum_{i=1}^n \mathcal{A}_i(\widetilde{x}_i^k) - c)]\rangle \geq 0. \end{cases}$$

Defining

$$\widetilde{\lambda}^k = \lambda^k - \sigma\left(\mathcal{A}_1(\widetilde{x}_1^k) + \sum_{i=2}^{n}\mathcal{A}_i(x_i^k) - c\right), \tag{15}$$

since the variable $\widehat{x}_i$ $(i = 2, \cdots, n-1)$ are intermediate in the iteration, it is only required to study the following inequality:

$$\begin{cases} \theta_1(x_1) - \theta_1(\widetilde{x}_1^k) + \left\langle x_1 - \widetilde{x}_1^k, -\mathcal{A}_1^*(\widetilde{\lambda}^k)\right\rangle \geq 0, \quad \forall x_1 \in \mathcal{X}_1, \\ \text{for} \quad j = n, \cdots, 3, \\ \theta_j(x_j) - \theta_j(\widetilde{x}_j^k) + \langle x_j - \widetilde{x}_j^k\,, \\ \qquad -\mathcal{A}_j^*[\widetilde{\lambda}^k + \sigma\sum_{i=2}^{j-1}\mathcal{A}_i(\widehat{x}_i^k - x_i^k) + \sigma\sum_{i=j}^{n}\mathcal{A}_i(\widetilde{x}_i^k - x_i^k)]\Big\rangle \geq 0, \quad \forall x_j \in \mathcal{X}_j, \\ \theta_2(x_2) - \theta_2(\widetilde{x}_2^k) + \left\langle x_2 - \widetilde{x}_2^k, -\mathcal{A}_2^*[\widetilde{\lambda}^k + \sigma\sum_{i=2}^{n}\mathcal{A}_i(\widetilde{x}_i^k - x_i^k)]\right\rangle \geq 0, \quad \forall x_2 \in \mathcal{X}_2. \end{cases} \tag{16}$$

Under Assumption 1, the solution of the subproblem with the variable x_i $(i = 2, \cdots, n-1)$ can be easily expressed, which will simplify our analysis. Since these subproblems are convex for its variables, by using the first-order optimality condition, we have

$$\psi_i(\widehat{x}_i^k) + b_i - \mathcal{A}_i^*[\lambda^k - \sigma(\mathcal{A}_1(\widetilde{x}_1^k) + \sum_{l=2}^{i}\mathcal{A}_l(\widehat{x}_l^k) + \sum_{l=i+1}^{n}\mathcal{A}_l(x_l^k) - c)] = 0, \quad i = 2, \cdots, n-1,$$

$$\psi_i(\widetilde{x}_i^k) + b_i - \mathcal{A}_i^*[\lambda^k - \sigma(\mathcal{A}_1(\widetilde{x}_1^k) + \sum_{l=2}^{i-1}\mathcal{A}_l(\widehat{x}_l^k) + \sum_{l=i}^{n}\mathcal{A}_l(\widetilde{x}_l^k) - c)] = 0, \quad i = n-1, \cdots, 3,$$

and

$$\psi_2(\widetilde{x}_2^k) + b_2 - \mathcal{A}_2^*[\lambda^k - \sigma(\sum_{l=1}^{n}\mathcal{A}_l(\widetilde{x}_l^k) - c)] = 0.$$

Therefore, we deduce

$$(\psi_i + \sigma\mathcal{A}_i^*\mathcal{A}_i)(\widehat{x}_i^k - \widetilde{x}_i^k) = \sigma\mathcal{A}_i^*\sum_{l=i+1}^{n}\mathcal{A}_l(\widetilde{x}_l^k - x_l^k)$$

for $i = 2, \cdots, n-1$.

From Assumption 1, the operator $\psi_i + \sigma\mathcal{A}_i^*\mathcal{A}_i$ is invertible for any $\sigma > 0$. Then, we have

$$\begin{aligned} \mathcal{A}_i(\widehat{x}_i^k - x_i^k) &= \mathcal{A}_i(\widehat{x}_i^k - \widetilde{x}_i^k) + \mathcal{A}_i(\widetilde{x}_i^k - x_i^k) \\ &= \mathcal{P}_i\sum_{l=i+1}^{n}\mathcal{A}_l(\widetilde{x}_l^k - x_l^k) + \mathcal{A}_i(\widetilde{x}_i^k - x_i^k), \end{aligned} \tag{17}$$

for $i = 2, \cdots, n-1$, where

$$\mathcal{P}_i = \sigma\mathcal{A}_i(\psi_i + \sigma\mathcal{A}_i^*\mathcal{A}_i)^{-1}\mathcal{A}_i^*, \tag{18}$$

and the operator ψ_i is defined as in (5).

Substituting (17) into (16), we can eliminate the intermediate variables $\widehat{x}_i^k \ \ (i = 2, \cdots, n-1)$ and obtain

$$\begin{cases} \theta_1(x_1) - \theta_1(\widetilde{x}_1^k) + \left\langle x_1 - \widetilde{x}_1^k, -\mathcal{A}_1^*(\widetilde{\lambda}^k) \right\rangle \geq 0, \forall x_1 \in \mathcal{X}_1, \\ \theta_2(x_2) - \theta_2(\widetilde{x}_2^k) + \left\langle x_2 - \widetilde{x}_2^k, -\mathcal{A}_2^*[\widetilde{\lambda}^k + \sigma \sum_{i=2}^n \mathcal{A}_i(\widetilde{x}_i^k - x_i^k)] \right\rangle \geq 0, \forall x_2 \in \mathcal{X}_2, \\ \text{for} \ \ j = 3, \cdots, n, \\ \theta_j(x_j) - \theta_j(\widetilde{x}_j^k) + \langle x_j - \widetilde{x}_j^k \ , \\ \qquad -\mathcal{A}_j^*[\widetilde{\lambda}^k + \sigma \sum_{i=2}^{j-1} \left(\mathcal{P}_i \sum_{l=i+1}^n \mathcal{A}_l(\widetilde{x}_l^k - x_l^k) + \mathcal{A}_i(\widetilde{x}_i^k - x_i^k)\right) + \sigma \sum_{i=j}^n \mathcal{A}_i(\widetilde{x}_i^k - x_i^k)] \right\rangle \geq 0. \end{cases} \tag{19}$$

Lemma 1 *Let $\widetilde{u}^k = (\widetilde{x}_{(1:n)}^k)^T$ be generated by PCB-ADMM from the given $w^k = (x_{(1:n)}^k, \lambda^k)^T$, then for any $w \in \mathcal{W}$, we have*

$$\theta(u) - \theta(\widetilde{u}^k) + \langle w - \widetilde{w}^k, \mathcal{F}(\widetilde{w}^k) \rangle \geq \langle v - \widetilde{v}^k, \mathcal{Q}(v^k - \widetilde{v}^k) \rangle, \tag{20}$$

where

$$\mathcal{Q} = \begin{pmatrix} \sigma\mathcal{A}_2^*\mathcal{A}_2 & \sigma\mathcal{A}_2^*\mathcal{A}_3 & \sigma\mathcal{A}_2^*\mathcal{A}_4 & \cdots & \sigma\mathcal{A}_2^*\mathcal{A}_n & 0 \\ \sigma\mathcal{A}_3^*\mathcal{A}_2 & \sigma\mathcal{A}_3^*(\mathcal{I}+\mathcal{P}_2)\mathcal{A}_3 & \sigma\mathcal{A}_3^*(\mathcal{I}+\mathcal{P}_2)\mathcal{A}_4 & \cdots & \sigma\mathcal{A}_3^*(\mathcal{I}+\mathcal{P}_2)\mathcal{A}_n & 0 \\ \sigma\mathcal{A}_4^*\mathcal{A}_2 & \sigma\mathcal{A}_4^*(\mathcal{I}+\mathcal{P}_2)\mathcal{A}_3 & \sigma\mathcal{A}_4^*(\mathcal{I}+\mathcal{P}_2+\mathcal{P}_3)\mathcal{A}_4 & \cdots & \sigma\mathcal{A}_4^*(\mathcal{I}+\mathcal{P}_2+\mathcal{P}_3)\mathcal{A}_n & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \sigma\mathcal{A}_n^*\mathcal{A}_2 & \sigma\mathcal{A}_n^*(\mathcal{I}+\mathcal{P}_2)\mathcal{A}_3 & \sigma\mathcal{A}_n^*(\mathcal{I}+\mathcal{P}_2+\mathcal{P}_3)\mathcal{A}_4 & \cdots & \sigma\mathcal{A}_n^*(\mathcal{I}+\sum_{i=2}^{n-1}\mathcal{P}_i)\mathcal{A}_n & 0 \\ -\mathcal{A}_2 & -\mathcal{A}_3 & -\mathcal{A}_4 & \cdots & -\mathcal{A}_n & \frac{1}{\sigma}\mathcal{I} \end{pmatrix}. \tag{21}$$

Proof. From the definition of $\widetilde{\lambda}^k$ in (15), we have

$$\sum_{i=1}^n \mathcal{A}_i(\widetilde{x}_i^k) - c = \sum_{i=2}^n \mathcal{A}_i(\widetilde{x}_i^k - x_i^k) - \frac{1}{\sigma}(\widetilde{\lambda}^k - \lambda^k),$$

namely

$$\left\langle \lambda - \widetilde{\lambda}^k, \quad \sum_{i=1}^n \mathcal{A}_i(\widetilde{x}_i^k) - c - \sum_{i=2}^n \mathcal{A}_i(\widetilde{x}_i^k - x_i^k) + \frac{1}{\sigma}(\widetilde{\lambda}^k - \lambda^k) \right\rangle = 0, \quad \forall \lambda \in \mathbb{R}^n.$$

Together with (19) and the definition of $\mathcal{F}(w)$ in (8), we can get the result easily. □

Using $\widetilde{\lambda}^k$ defined in (15), $\widehat{\lambda}^k$ updated can be represented as

$$\widehat{\lambda}^k \quad = \quad \lambda^k - \sigma\left(\sum_{i=1}^n \mathcal{A}_i(\widetilde{x}_i^k) - c\right) = \lambda^k - \left((\lambda^k - \widetilde{\lambda}^k - \sigma\sum_{i=2}^n \mathcal{A}_i(x_i^k - \widetilde{x}_i^k)\right).$$

Thus, the correction $\lambda^{k+1} = \lambda^k - \alpha(\lambda^k - \widehat{\lambda}^k)$ can be rewritten as

$$\lambda^{k+1} \quad = \quad \lambda^k - \alpha(-\sigma\mathcal{A}_2, \quad \cdots, \quad -\sigma\mathcal{A}_n, \mathcal{I})(v^k - \widetilde{v}^k).$$

Combining it with the correcting of $x_{(2:n)}^k$ in (14), the correction step can be expressed as

$$\begin{cases} x_1^{k+1} = \widetilde{x}_1^k \\ v^{k+1} = v^k - \alpha\mathcal{M}(v^k - \widetilde{v}^k) \end{cases}, \tag{22}$$

where

$$\mathcal{M} = \begin{pmatrix} \mathcal{I} & 0 & \cdots & 0 & 0 \\ 0 & \mathcal{I} & \ddots & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathcal{I} & 0 \\ -\sigma\mathcal{A}_2 & -\sigma\mathcal{A}_3 & \cdots & -\sigma\mathcal{A}_n & \mathcal{I} \end{pmatrix}. \tag{23}$$

In the next section, we analyse the convergence of PCB-ADMM (Algorithm 1) with the use of the correction step (22).

3 Convergence analysis

In [8], He et.al. proposed an ADMM-based algorithm for solving (1), and defined

$$\mathcal{H} = \mathcal{Q}\mathcal{M}^{-1}, \quad \mathcal{G} = \mathcal{Q} + \mathcal{Q}^* - \alpha\mathcal{M}^*\mathcal{H}\mathcal{M} \tag{24}$$

with using $\mathcal{Q}$ and $\mathcal{M}$. Based on these definitions, Convergence Condition was given in [8]:

> if $\mathcal{Q} + \mathcal{Q}^*$, $\mathcal{H}$ and $\mathcal{G}$ are positive definite, the ADMM-based algorithm is convergent.

Since PCB-ADMM with $\alpha \in (0, 1)$ is a special ADMM-based algorithm, we can obtain its convergence by checking this Convergence Condition. For PCB-ADMM with $\alpha = 1$, the Convergence Condition is not satisfied. However, under an extra condition that the operator $\mathcal{A}_2$ is invertible, we can obtain its convergence. In the following, we use the notation $\|v\|^2_{\mathcal{M}} := \langle v, \mathcal{M}v\rangle$ for a given positive semidefinite operator $\mathcal{M}$.

Lemma 2 *For the operator $\mathcal{P}_i (i = 2, \cdots, n)$ defined in (18), let*

$$\widetilde{\mathcal{P}}_i = \mathcal{A}_i^* \mathcal{P}_i \mathcal{A}_i, \tag{25}$$

then the operator $\widetilde{\mathcal{P}}_i$ is positive definite.

Proof. From the definition of $\mathcal{P}_i$ in (18), we have

$$\widetilde{\mathcal{P}}_i = \mathcal{A}_i^* \mathcal{A}_i \left(\sigma(\psi_i + \sigma\mathcal{A}_i^*\mathcal{A}_i)^{-1}\right) \mathcal{A}_i^*\mathcal{A}_i.$$

Since the operator $\mathcal{A}_i^*\mathcal{A}_i$ is inevitable from Assumption 1, it implies that $\widetilde{\mathcal{P}}_i$ is congruent to $\sigma(\psi_i + \sigma\mathcal{A}_i^*\mathcal{A}_i)^{-1}$. Thus, the results can be obtained by the positive definiteness of $\sigma(\psi_i + \sigma\mathcal{A}_i^*\mathcal{A}_i)^{-1}$. □

Lemma 3 *Under Assumption 1, if the operators $\mathcal{Q}$ and $\mathcal{M}$ are defined as in (21) and (23), then we have*

(i). the operator $\mathcal{Q} + \mathcal{Q}^*$ is positive definite;

(ii). the operator $\mathcal{H}$ is positive definite, where $\mathcal{H} = \mathcal{Q}\mathcal{M}^{-1}$.

Proof. (i). It follows from (21) that we can obtain $\mathcal{Q}+\mathcal{Q}^*$ easily, and rewrite it as $\mathcal{Q}_1+\mathcal{Q}_2$, where

$$\mathcal{Q}_1 = \begin{pmatrix} \sigma\mathcal{A}_2^*\mathcal{A}_2 & \sigma\mathcal{A}_2^*\mathcal{A}_3 & \sigma\mathcal{A}_2^*\mathcal{A}_4 & \cdots & \sigma\mathcal{A}_2^*\mathcal{A}_n & -\mathcal{A}_2^* \\ \sigma\mathcal{A}_3^*\mathcal{A}_2 & \sigma\mathcal{A}_3^*\mathcal{A}_3 & \sigma\mathcal{A}_3^*\mathcal{A}_4 & \cdots & \sigma\mathcal{A}_3^*\mathcal{A}_n & -\mathcal{A}_3^* \\ \sigma\mathcal{A}_4^*\mathcal{A}_2 & \sigma\mathcal{A}_4^*\mathcal{A}_3 & \sigma\mathcal{A}_4^*\mathcal{A}_4 & \cdots & \sigma\mathcal{A}_4^*\mathcal{A}_n & -\mathcal{A}_4^* \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \sigma\mathcal{A}_n^*\mathcal{A}_2 & \sigma\mathcal{A}_n^*\mathcal{A}_3 & \sigma\mathcal{A}_n^*\mathcal{A}_4 & \cdots & \sigma\mathcal{A}_n^*\mathcal{A}_n & -\mathcal{A}_n^* \\ -\mathcal{A}_2 & -\mathcal{A}_3 & -\mathcal{A}_4 & \cdots & -\mathcal{A}_n & \frac{1}{\sigma}\mathcal{I} \end{pmatrix},$$

$$\mathcal{Q}_2 = \begin{pmatrix} \sigma\mathcal{A}_2^*\mathcal{A}_2 & \sigma\mathcal{A}_2^*\mathcal{A}_3 & \sigma\mathcal{A}_2^*\mathcal{A}_4 & \cdots & \sigma\mathcal{A}_2^*\mathcal{A}_n & 0 \\ \sigma\mathcal{A}_3^*\mathcal{A}_2 & \sigma\mathcal{A}_3^*(\mathcal{I}+2\mathcal{P}_2)\mathcal{A}_3 & \sigma\mathcal{A}_3^*(\mathcal{I}+2\mathcal{P}_2)\mathcal{A}_4 & \cdots & \sigma\mathcal{A}_3^*(\mathcal{I}+2\mathcal{P}_2)\mathcal{A}_n & 0 \\ \sigma\mathcal{A}_4^*\mathcal{A}_2 & \sigma\mathcal{A}_4^*(\mathcal{I}+2\mathcal{P}_2)\mathcal{A}_3 & \sigma\mathcal{A}_4^*(\mathcal{I}+2(\mathcal{P}_2+\mathcal{P}_3))\mathcal{A}_4 & \cdots & \sigma\mathcal{A}_4^*(\mathcal{I}+2(\mathcal{P}_2+\mathcal{P}_3))\mathcal{A}_n & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \sigma\mathcal{A}_n^*\mathcal{A}_2 & \sigma\mathcal{A}_n^*(\mathcal{I}+2\mathcal{P}_2)\mathcal{A}_3 & \sigma\mathcal{A}_n^*(\mathcal{I}+2(\mathcal{P}_2+\mathcal{P}_3))\mathcal{A}_4 & \cdots & \sigma\mathcal{A}_n^*(\mathcal{I}+2\sum_{i=2}^{n-1}\mathcal{P}_i)\mathcal{A}_n & 0 \\ 0 & 0 & 0 & \cdots & 0 & \frac{1}{\sigma}\mathcal{I} \end{pmatrix}.$$

Notice that from the structure of $\mathcal{Q}_1$, we have $\mathcal{Q}_1 \succeq 0$, then $\mathcal{Q}_1+\mathcal{Q}_2 \succeq \mathcal{Q}_2$. In the following, we will show that, $\mathcal{Q}_2$ is positive definite by deduction.

For any $v \in \mathcal{V}$, we deduce that

$$\langle v, \mathcal{Q}_2(v)\rangle = \sigma\left\|\sum_{i=2}^{n}\mathcal{A}_i(x_i)\right\|^2 + 2\sigma\sum_{j=3}^{n}\left\|\sum_{i=j}^{n}\mathcal{A}_i(x_i)\right\|_{\mathcal{P}_{j-1}}^2 + \frac{1}{\sigma}\|\lambda\|^2 \geq 0.$$

If $\langle v, \mathcal{Q}_2(v)\rangle = 0$, it means

$$\lambda = 0, \tag{26}$$

$$\left\|\sum_{i=2}^{n}\mathcal{A}_i(x_i)\right\|^2 = 0, \tag{27}$$

$$\left\|\sum_{i=j}^{n}\mathcal{A}_i(x_i)\right\|_{\mathcal{P}_{j-1}}^2 = 0, \quad \text{for} \quad j = 3, \cdots, n. \tag{28}$$

Note that from (27), we have $\sum_{i=2}^{n}\mathcal{A}_i(x_i) = 0$. Together with (28), then we have

$$\left.\begin{array}{l} \sum_{i=3}^{n}\mathcal{A}_i(x_i) = -\mathcal{A}_2(x_2) \\ \|\sum_{i=3}^{n}\mathcal{A}_i(x_i)\|_{\mathcal{P}_2}^2 = 0 \end{array}\right\} \Rightarrow \|\mathcal{A}_2(x_2)\|_{\mathcal{P}_2}^2 = 0, \tag{29}$$

From (25) and $\widetilde{\mathcal{P}}_i$ being positive definite, we get

$$\|\mathcal{A}_2(x_2)\|_{\mathcal{P}_2}^2 = 0 \Rightarrow \|x_2\|_{\widetilde{\mathcal{P}}_2}^2 = 0 \Rightarrow x_2 = 0.$$

Substituting (29) into (27) and (28), we have

$$\left.\begin{array}{l} \sum_{i=4}^{n}\mathcal{A}_i(x_i) = -\mathcal{A}_3(x_3) \\ \|\sum_{i=4}^{n}\mathcal{A}_i(x_i)\|_{\mathcal{P}_3}^2 = 0 \end{array}\right\} \Rightarrow \|\mathcal{A}_3(x_3)\|_{\mathcal{P}_3}^2 = 0 \Rightarrow \|x_3\|_{\widetilde{\mathcal{P}}_3}^2 = 0 \Rightarrow x_3 = 0.$$

Similarly, we can obtain $x_4 = x_5 = \cdots = x_n = 0$, which implies that, $\langle v, \mathcal{Q}_2(v)\rangle > 0$ for any $v \neq 0$. Namely $\mathcal{Q}_2$ is positive definite. Thus, the operator $\mathcal{Q}+\mathcal{Q}^*$ is positive definite.

(ii). By using the computation of the operator $\mathcal{Q}$ and $\mathcal{M}$, we obtain

$$\mathcal{H} = \sigma \begin{pmatrix} \mathcal{A}_2^*\mathcal{A}_2 & \mathcal{A}_2^*\mathcal{A}_3 & \mathcal{A}_2^*\mathcal{A}_4 & \cdots & \mathcal{A}_2^*\mathcal{A}_n & 0 \\ \mathcal{A}_3^*\mathcal{A}_2 & \mathcal{A}_3^*(\mathcal{I}+\mathcal{P}_2)\mathcal{A}_3 & \mathcal{A}_3^*(\mathcal{I}+\mathcal{P}_2)\mathcal{A}_4 & \cdots & \mathcal{A}_3^*(\mathcal{I}+\mathcal{P}_2)\mathcal{A}_n & 0 \\ \mathcal{A}_4^*\mathcal{A}_2 & \mathcal{A}_4^*(\mathcal{I}+\mathcal{P}_2)\mathcal{A}_3 & \mathcal{A}_4^*(\mathcal{I}+\mathcal{P}_2+\mathcal{P}_3)\mathcal{A}_4 & \cdots & \mathcal{A}_4^*(\mathcal{I}+\mathcal{P}_2+\mathcal{P}_3)\mathcal{A}_n & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathcal{A}_n^*\mathcal{A}_2 & \mathcal{A}_n^*(\mathcal{I}+\mathcal{P}_2)\mathcal{A}_3 & \mathcal{A}_n^*(\mathcal{I}+\mathcal{P}_2+\mathcal{P}_3)\mathcal{A}_4 & \cdots & \mathcal{A}_n^*(\mathcal{I}+\sum_{i=2}^{n-1}\mathcal{P}_i)\mathcal{A}_n & 0 \\ 0 & 0 & 0 & \cdots & 0 & \frac{1}{\sigma^2}\mathcal{I} \end{pmatrix}. \tag{30}$$

For any $v \in \mathcal{V}$, we deduce that

$$\langle v, \mathcal{H}(v)\rangle \quad = \quad \sigma \sum_{j=2}^{n} \left\| \sum_{i=j}^{n} \mathcal{A}_i(x_i) \right\|^2_{\mathcal{P}_{j-1}} + \frac{1}{\sigma}\|\lambda\|^2 \geq 0,$$

where $\mathcal{P}_1 = \mathcal{I}$. By using the same strategy as in the proof of (i), we can obtain the results. □

Now, for $\alpha \in (0, 1]$ we define

$$\mathcal{G}(\alpha) := \mathcal{Q} + \mathcal{Q}^* - \alpha\mathcal{M}^*\mathcal{H}\mathcal{M}, \tag{31}$$

then

$$\mathcal{G}(\alpha) = 2\sigma \begin{pmatrix} \beta\mathcal{A}_2^*\mathcal{A}_2 & \beta\mathcal{A}_2^*\mathcal{A}_3 & \cdots & \beta\mathcal{A}_2^*\mathcal{A}_n & -\beta\mathcal{A}_2^* \\ \beta\mathcal{A}_3^*\mathcal{A}_2 & \mathcal{A}_3^*(\beta\mathcal{I}+\gamma\mathcal{P}_2)\mathcal{A}_3 & \cdots & \mathcal{A}_3^*(\beta\mathcal{I}+\gamma\mathcal{P}_2)\mathcal{A}_n & -\beta\mathcal{A}_3^* \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \beta\mathcal{A}_n^*\mathcal{A}_2 & \mathcal{A}_n^*(\beta\mathcal{I}+\gamma\mathcal{P}_2)\mathcal{A}_3 & \cdots & \mathcal{A}_n^*(\beta\mathcal{I}+\gamma\sum_{i=2}^{n-1}\mathcal{P}_i)\mathcal{A}_n & -\beta\mathcal{A}_n^* \\ -\beta\mathcal{A}_2 & -\beta\mathcal{A}_3 & \cdots & -\beta\mathcal{A}_n & \frac{2-\alpha}{\sigma^2}\mathcal{I} \end{pmatrix}. \tag{32}$$

where $\beta = 1 - \alpha$ and $\gamma = 1 - \frac{1}{2}\alpha$. Obviously, if $\alpha = 0$, $\mathcal{G}(0) = \mathcal{Q} + \mathcal{Q}^* \succ 0$, else if $\alpha = 1$, then $\beta = 0$ and $\gamma = \frac{1}{2}$. For the case of $\alpha = 1$, we have

$$\mathcal{G}(1) = \sigma \begin{pmatrix} 0 & 0 \\ 0 & \mathcal{G}_0 \end{pmatrix} \succeq 0, \quad \text{where} \quad \mathcal{G}_0 = \begin{pmatrix} \mathcal{A}_3^*(\mathcal{P}_2)\mathcal{A}_3 & \cdots & \mathcal{A}_3^*(\mathcal{P}_2)\mathcal{A}_n & 0 \\ \vdots & \ddots & \vdots & \vdots \\ \mathcal{A}_n^*(\mathcal{P}_2)\mathcal{A}_3 & \cdots & \mathcal{A}_n^*(\sum_{i=2}^{n-1}\mathcal{P}_i)\mathcal{A}_n & 0 \\ 0 & \cdots & 0 & \frac{1}{\sigma^2}\mathcal{I} \end{pmatrix}. \tag{33}$$

Thus, we deduce that

$$\mathcal{G}(\alpha) = (1-\alpha)(\mathcal{Q} + \mathcal{Q}^*) + \alpha\mathcal{G}(1).$$

Then, $\mathcal{Q} + \mathcal{Q}^* \succ 0$ and $\mathcal{G}(1) \succeq 0$ imply that $\mathcal{G}(\alpha)$ is positive definite for $\alpha \in (0, 1)$.

Theorem 1 *For the sequence $\{v^k\}$ generated by PCB-ADMM with $\alpha \in (0, 1)$, it converges to some $\widetilde{v}$ which belongs to $\mathcal{V}^*$.*

Proof. From Lemma 3, for any $\alpha \in (0, 1)$, the operators $\mathcal{Q} + \mathcal{Q}^*$, $\mathcal{H}$ and $\mathcal{G}(\alpha)$ are positive definite. Thus, the Convergence Condition introduced in [8] is satisfied, and then the convergence of PCB-ADMM with $\alpha \in (0, 1)$ can be obtained. □

In sequel, firstly we show some properties of the sequence $\{w^k\}$ generated by PCB-ADMM, and then discuss the convergence of PCB-ADMM with $\alpha = 1$.

Lemma 4 *For the sequence $\{w^k\}$ generated by PCB-ADMM with $\alpha \in (0,1]$, we have*

$$\alpha\{\theta(u) - \theta(\widetilde{u}^k) + \langle w - \widetilde{w}^k, \mathcal{F}(\widetilde{w}^k)\rangle\} \geq \frac{1}{2}(\|v - v^{k+1}\|_{\mathcal{H}}^2 - \|v - v^k\|_{\mathcal{H}}^2) + \frac{\alpha}{2}\|v^k - \widetilde{v}^k\|_{\mathcal{G}(\alpha)}^2$$

for any $w \in \mathcal{W}$, where $\mathcal{H} = \mathcal{Q}\mathcal{M}^{-1}$.

Proof. From Lemma 1 and $\mathcal{Q} = \mathcal{H}\mathcal{M}$, for any $w \in \mathcal{W}$ we have

$$\begin{aligned} \theta(u) - \theta(\widetilde{u}^k) + \langle w - \widetilde{w}^k, \mathcal{F}(\widetilde{w}^k)\rangle &\geq \langle v - \widetilde{v}^k, \mathcal{H}\mathcal{M}(v^k - \widetilde{v}^k)\rangle \\ &= \frac{1}{\alpha}\langle v - \widetilde{v}^k, \mathcal{H}(v^k - v^{k+1})\rangle \end{aligned} \tag{34}$$

where the equality above comes from (22). Since $\mathcal{H}$ is symmetric, we can obtain

$$\begin{aligned} \langle v - \widetilde{v}^k, \mathcal{H}(v^k - v^{k+1})\rangle &= \frac{1}{2}(\|v - v^{k+1}\|_{\mathcal{H}}^2 - \|v - v^k\|_{\mathcal{H}}^2) \\ &\quad + \frac{1}{2}(\|v^k - \widetilde{v}^k\|_{\mathcal{H}}^2 - \|v^{k+1} - \widetilde{v}^k\|_{\mathcal{H}}^2). \end{aligned} \tag{35}$$

In addition, using the definition of $\mathcal{G}(\alpha)$ in (32), we have

$$\|v^k - \widetilde{v}^k\|_{\mathcal{H}}^2 - \|v^{k+1} - \widetilde{v}^k\|_{\mathcal{H}}^2 = \alpha\|v^k - \widetilde{v}^k\|_{\mathcal{G}(\alpha)}^2. \tag{36}$$

Substituting (35) and (36) into (34), we get the result. □

Lemma 5 *[8, Theorem 4.2] For the sequence $\{v^k\}$ generated by PCB-ADMM, we have*

$$\|v^{k+1} - v^*\|_{\mathcal{H}}^2 \leq \|v^k - v^*\|_{\mathcal{H}}^2 - \alpha\|v^k - \widetilde{v}^k\|_{\mathcal{G}(\alpha)}^2, \quad \forall v^* \in \mathcal{V}^*. \tag{37}$$

In the following, we show the positive definiteness of the operator $\mathcal{G}_0$ and discuss the relation between λ^{k+1} and x_2^{k+1}, which is important for obtaining the convergence of PCB-ADMM with $\alpha = 1$.

Lemma 6 *Suppose that the operator $\mathcal{A}_2$ is invertible, then*

(i). the operator $\mathcal{G}_0$ defined in (33) is positive definite;

(ii). for the sequence $\{v^k\}$ generated by PCB-ADMM with $\alpha = 1$, we have

$$\lambda^{k+1} = (\mathcal{A}_2^*)^{-1}(\psi_2(x_2^{k+1}) + b_2). \tag{38}$$

Proof. (i). For any $z = (x_{(3:n)}, \lambda)^T$, from the definition of $\mathcal{G}_0$, we obtain

$$\langle z, \mathcal{G}_0 z\rangle = \left\|\sum_{i=3}^n \mathcal{A}_i(x_i)\right\|_{\mathcal{P}_2}^2 + \sum_{j=4}^n \left\|\sum_{i=j}^n \mathcal{A}_i(x_i)\right\|_{\mathcal{P}_{j-1}}^2 + \frac{1}{\sigma^2}\|\lambda\|^2 \geq 0.$$

If $\langle z, \mathcal{G}_0 z\rangle = 0$, then $\lambda = 0$ and $\left\|\sum_{i=j}^n \mathcal{A}_i(x_i)\right\|_{\mathcal{P}_{j-1}}^2 = 0$ for $j = 3, \cdots, n$. Since $\mathcal{A}_2$ is invertible, then $\mathcal{P}_2$ is positive definite and

$$\left\|\sum_{i=3}^n \mathcal{A}_i(x_i)\right\|^2 = 0 \Rightarrow \sum_{i=3}^n \mathcal{A}_i(x_i) = 0.$$

Combining it with $\|\sum_{i=4}^{n}\mathcal{A}_i(x_i)\|^2_{\mathcal{P}_3}=0$, we have $\|\mathcal{A}_3(x_3)\|^2_{\mathcal{P}_3}=\|x_3\|^2_{\widetilde{\mathcal{P}}_3}=0$. Since the operator $\widetilde{\mathcal{P}}_3$ is positive definite from Lemma 2, then we get $x_3=0$. With easy, we can obtain $x_4=x_5=\cdots=x_n=0$ from $\langle z,\mathcal{G}_0 z\rangle=0$, which implies $\mathcal{G}_0$ is positive definite.

(ii). From the first-order optimality condition for the subproblem with the variable x_2, we have

$$(\psi_2+\sigma\mathcal{A}_2^*\mathcal{A}_2)(x_2^{k+1})=\mathcal{A}_2^*[\lambda^k-\sigma(\mathcal{A}_1(x_1^{k+1})+\sum_{i=3}^{n}\mathcal{A}_i(x_i^{k+1})-c)]-b_2. \tag{39}$$

By the correction step, the updating of λ^{k+1} can be written as

$$\lambda^{k+1}=\lambda^k-\sigma(\sum_{i=1}^{n}\mathcal{A}_i(x_i^{k+1})-c), \tag{40}$$

Together (39) with (40), we obtain

$$\begin{aligned}\mathcal{A}_2^*(\lambda^{k+1}) &= \mathcal{A}_2^*[\lambda^k-\sigma(\mathcal{A}_1(x_1^{k+1})+\sum_{i=3}^{n}\mathcal{A}_i(x_i^{k+1})-c)]-\sigma\mathcal{A}_2^*\mathcal{A}_2(x_2^{k+1})\\ &= (\psi_2+\sigma\mathcal{A}_2^*\mathcal{A}_2)(x_2^{k+1})+b_2-\sigma\mathcal{A}_2^*\mathcal{A}_2(x_2^{k+1})\\ &= \psi_2(x_2^{k+1})+b_2.\end{aligned}$$

Since the operator $\mathcal{A}_2$ is invertible, then $\mathcal{A}_2^*$ is invertible and we can obtain (38). This completes the proof. □

Theorem 2 *When the operator $\mathcal{A}_2$ is invertible, the sequence $\{v^k\}$ generated by PCB-ADMM with $\alpha=1$ converges to some $\widetilde{v}$ which belongs to $\mathcal{V}^*$.*

Proof. If $\alpha=1$, we know that $\mathcal{Q}+\mathcal{Q}^*$ and $\mathcal{H}$ are positive definite, but $\mathcal{G}(1)$ is positive semidefinite. Namely, the Convergence Condition is not satisfied.

Since $\|v^k-\widetilde{v}^k\|^2_{\mathcal{G}(1)}\geq 0$, combining it with (37), we have

$$\|v^{k+1}-v^*\|^2_{\mathcal{H}}\leq\|v^k-v^*\|^2_{\mathcal{H}},\quad\forall v^*\in\mathcal{V}^*.$$

Therefore, the sequence $\{v^k\}$ is bounded. From Lemma 5, we get

$$0\leq\|v^k-\widetilde{v}^k\|^2_{\mathcal{G}(1)}\leq\|v^k-v^*\|^2_{\mathcal{H}}-\|v^{k+1}-v^*\|^2_{\mathcal{H}},\quad\forall v^*\in\mathcal{V}^*.$$

By taking limit, we get $\lim_{k\to\infty}\|v^k-\widetilde{v}^k\|^2_{\mathcal{G}(1)}=0$, which together with (33) means,

$$\lim_{k\to\infty}\|z^k-\widetilde{z}^k\|^2_{\mathcal{G}_0}=0,$$

where $z^k=(x^k_{(3:n)},\lambda^k)^T$ and $\widetilde{z}^k=(\widetilde{x}^k_{(3:n)},\widetilde{\lambda}^k)^T$. From the results in Lemma 6, the operator $\mathcal{G}_0$ is positive definite, then the sequence $\{\widetilde{z}^k\}$ is bounded because of the boundness of the sequence $\{z^k\}$. Thus, $\{z^k\}$ and $\{\widetilde{z}^k\}$ lie in a compact set and must have limit point, say

$$\lim_{k\to\infty}\|z^k-\widetilde{z}^k\|^2=\|\widehat{z}-\widetilde{z}\|^2=0, \tag{41}$$

where $\widehat{z}$ and $\widetilde{z}$ can be any limit point of the sequence $\{z^k\}$ and $\{\widetilde{z}^k\}$, respectively. Namely, there exists a subsequence $\{k_j\}_{j=1}^{\infty}$ such that

$$\lim_{j\to\infty} z^{k_j} = \widehat{z} = \widetilde{z} = \lim_{j\to\infty} \widetilde{z}^{k_j}. \tag{42}$$

In addition, it follows from (15) that

$$\lim_{j\to\infty} (\lambda^{k_j} - \widetilde{\lambda}^{k_j}) = \lim_{j\to\infty} [(\lambda^{k_j} - \lambda^{k_j+1}) + \sigma \sum_{i=2}^{n} \mathcal{A}_i(x_i^{k_j} - \widetilde{x}_i^{k_j})] = 0.$$

Using Lemma 6 and the continuity of ψ_2 and $\mathcal{A}_i$, we obtain

$$\begin{aligned} 0 &= \lim_{j\to\infty} \mathcal{A}_2^*[(\lambda^{k_j} - \lambda^{k_j+1}) + \sigma \sum_{i=2}^{n} \mathcal{A}_i(x_i^{k_j} - \widetilde{x}_i^{k_j})] \\ &= \lim_{j\to\infty} [(\psi_2 + \sigma\mathcal{A}_2^*\mathcal{A}_2)(x_2^{k_j} - \widetilde{x}_2^{k_j})] + \lim_{j\to\infty} \mathcal{A}_2^*[\sigma \sum_{i=3}^{n} \mathcal{A}_i(x_i^{k_j} - \widetilde{x}_i^{k_j})] \\ &= (\psi_2 + \sigma\mathcal{A}_2^*\mathcal{A}_2) \lim_{j\to\infty} (x_2^{k_j} - \widetilde{x}_2^{k_j}). \end{aligned}$$

Since $(\psi_2 + \sigma\mathcal{A}_2^*\mathcal{A}_2)$ is positive definite, then

$$\lim_{j\to\infty} (x_2^{k_j} - \widetilde{x}_2^{k_j}) = 0. \tag{43}$$

Combining it with (42), we have $\lim_{j\to\infty}(v^{k_j} - \widetilde{v}^{k_j}) = 0$. Moreover, it follows from Assumption 1, (14) and (15) that

$$x_1^{k+1} = \widetilde{x}_1^k = (\mathcal{A}_1^*\mathcal{A}_1)^{-1}\mathcal{A}_1^* \left(\frac{1}{\sigma}(\lambda^k - \widetilde{\lambda}^k) - \sum_{i=2}^{n} \mathcal{A}_i(x_i^k) + c \right).$$

Therefore, the sequence $\{x_1^k\}$ is also bounded and the subsequence $\{x_1^{k_j}\}$ is convergent. We set $\widetilde{x}_1 = \lim_{j\to\infty} x_1^{k_j}$.

For any $w \in \mathcal{W}$, from Lemma 1 we have

$$\theta(u) - \theta(\widetilde{u}^{k_j}) + \langle w - \widetilde{w}^{k_j}, \mathcal{F}(\widetilde{w}^{k_j})\rangle \geq \langle v - \widetilde{v}^{k_j}, \mathcal{Q}(v^{k_j} - \widetilde{v}^{k_j})\rangle, \quad \widetilde{w}^{k_j} \in \mathcal{W}.$$

Note that the continuity of $\theta(u)$ and $\mathcal{F}(w)$, it holds that

$$\widetilde{w} \in \mathcal{W}, \quad \theta(u) - \theta(\widetilde{u}) + \langle w - \widetilde{w}, \mathcal{F}(\widetilde{w})\rangle \geq 0.$$

The above variational inequality indicates that $\widetilde{w}$ is a solution point of the variational inequality (10). Since $\lim_{j\to\infty} v^{k_j} = \widehat{v} = \widetilde{v}$, then $\widetilde{v} \in \mathcal{V}^*$. Notice that (37), we have

$$\|v^{k+1} - \widetilde{v}\|_{\mathcal{H}}^2 \leq \|v^k - \widetilde{v}\|_{\mathcal{H}}^2, \tag{44}$$

then $\{v^k\}$ converges to $\widetilde{v}$. This completes the proof. □

Under the condition that the linear operator $\mathcal{A}_2$ is invertible, we obtain the convergence of PCB-ADMM with $\alpha = 1$. In this case, at least one invertible operator is required in all the operators $\mathcal{A}_i (i = 2, \cdots, n-1)$, unlike that in Assumption 1. In fact, this condition is easy to achievement in application. For instance, the linear constraint is $\sum_{i=1}^{n} x_i = c$ for the problem to recover the low-rank and sparse components of a given matrix, in which all the operators $\mathcal{A}_i = \mathcal{I}$ are invertible. In addition, Assumption 1 shows the semidefinite operator φ is surjective for the dual of the CQSDP with or without nonnegative constraint, which implies that φ is invertible, see the problem (46). Namely, Assumption 1 implies that φ is invertible, no extra condition is required for PCB-ADMM with $\alpha = 1$ to solve the dual of CQSDP problems.

4 Numerical results

We use PCB-ADMM for solving two types of important problems: convex quadratic semidefinite programming (CQSDP) with nonnegative constraints [11] and low patch-rank image decomposition [26, 27, 28, 29], and report the numerical results. To further illustrate its numerical efficiency, we compare the PCB-ADMM numerically with some other efficient methods for solving these problems. We denote the random number generator by *seed* for generating data again in MATLAB R2013B. All experiments are performed on an Intel(R) Core(TM) i5-4590 CPU@ 3.30 GHz PC with 8GB of RAM running on 64-bit Windows operating system.

4.1 Convex quadratic SDP

The CQSDP problem with nonnegative constraints has the following form:

$$\begin{array}{ll} \min & \frac{1}{2}\langle X, \varphi(X)\rangle + \langle C, X\rangle \\ s.t. & \mathcal{A}(X) = b, \\ & X \in \mathcal{S}^n_+, \quad X \in \mathcal{N}^n, \end{array} \tag{45}$$

where $\mathcal{S}^n_+$ is the cone of $n \times n$ symmetric and positive semi-definite matrices in the space of $n \times n$ symmetric matrices $\mathcal{S}^n$, endowed with the standard trace inner product $\langle \cdot, \cdot \rangle$ and the Frobenius norm $\|\cdot\|_F$. $C \in \mathcal{S}^n$, $b \in \mathbb{R}^m$ are given data. $\mathcal{N}^n$ is a nonempty simple closed convex set and

$$\mathcal{N}^n = \{X \in \mathcal{S}^n | \ X \geq 0\}.$$

$\mathcal{A} : \mathcal{S}^n \to \mathbb{R}^m$ is a linear map. The adjoint of $\mathcal{A}$ with respect to the standard inner product in $\mathcal{S}^n$ and $\mathbb{R}^m$ is denoted by $\mathcal{A}^*$. $\varphi : \mathcal{S}^n \to \mathcal{S}^n$ is a given self-adjoint positive semidefinite linear operator.

The dual of the problem (45) can be formulated as:

$$\begin{array}{ll} \min & \frac{1}{2}\langle X, \varphi(X)\rangle - b^T y + \delta_{\mathcal{S}^n_+}(Z) + \delta_{\mathcal{N}^n}(S) \\ s.t. & -\varphi(X) + \mathcal{A}^*(y) + Z + S = C, \end{array} \tag{46}$$

where $\delta_{\mathcal{C}}(Y)$ is the indicator function over the given set $\mathcal{C}$. Obviously, the dual problem (46) is a 4-block separable convex optimization problem with the objective $\frac{1}{2}\langle X, \varphi(X)\rangle$ being quadratic convex and $-b^T y$ linear, so we can apply PCB-ADMM to solve it. In this paper, we use the BCD cycle with the order $S \to X \to y \to Z \to y \to X$ for solving the subproblems.

Note that from Assumption 1, the linear operator φ should be positive definite. Therefore, we test the problems with three types of the positive definite operators φ in this section,

(a) $\varphi(X) = X$, as in the least squares SDP [30];

(b) $\varphi(X) = \frac{BX+XB}{2}$, where the matrix B is a random symmetric positive definite matrix.

(c) $\varphi(X) = WXW^T$ for a given positive-definite matrix $W \in \mathcal{S}^n$. The matrix W is generated as in [31] by

```
alpha=0.9; temp= orth(rand(n,n));
lambda1=400*rand(1,ceil(alpha*n))+100;
lambda2=99.99*rand(1,n-ceil(alpha*n))+0.01;
lambda=[lambda1 lambda2];
W = temp*diag(lambda)*temp';
```

From the code above, W has about 90% eigenvalues in the interval $[100, 500]$, and the rest of eigenvalues in the interval $[0.01, 100]$.

The data C and $\mathcal{A}$ in the tested problems are from the SDP relaxation $\theta_+(G)$ of the maximum stable set problem for a given graph G. We test the graph instances G considered as in [10].

The optimality conditions (KKT conditions) for the problems (45) and (46) can be written as follows:

$$\left.\begin{array}{c} \mathcal{A}(X) = b, \ -\varphi(X) + \mathcal{A}^*(y) + Z + S = C, \\ X \in \mathcal{S}^n_+, \ Z \in \mathcal{S}^n_+, \ \langle Z, X\rangle = 0, X \in \mathcal{N}^n, \ S \in \mathcal{N}^n, \ \langle S, X\rangle = 0. \end{array}\right\} \tag{47}$$

Therefore, we measure the accuracy of an approximate optimal solution (X, y, Z, S) for the primal problem and its dual by using the following relative residual:

$$\delta := \max \ \{\text{pinf}, \ \text{dinf}, \ p\delta_{\mathcal{S}^n_+}, \ p\delta_{\mathcal{N}^n}, \ d\delta_{\mathcal{S}^n_+}, \ d\delta_{\mathcal{N}^n}, \ \delta_{XZ}, \ \delta_{XS}\}, \tag{48}$$

where

$$\begin{array}{rcll} \text{pinf} & = & \dfrac{\|\mathcal{A}(X) - b\|_2}{1 + \|b\|_2}, & \text{dinf} = \dfrac{\|C + \varphi(X) - Z - \mathcal{A}^*(y)\|_F}{1 + \|C\|_F}, \\ p\delta_{\mathcal{S}^n_+} & = & \dfrac{\|\Pi_{\mathcal{S}^n_+}(-X)\|_F}{1 + \|X\|_F}, & p\delta_{\mathcal{N}^n} = \dfrac{\|\Pi_{\mathcal{N}^n}(-X)\|_F}{1 + \|X\|_F}, \\ d\delta_{\mathcal{S}^n_+} & = & \dfrac{\|\Pi_{\mathcal{S}^n_+}(-Z)\|_F}{1 + \|Z\|_F}, & d\delta_{\mathcal{N}^n} = \dfrac{\|\Pi_{\mathcal{N}^n}(-S)\|_F}{1 + \|S\|_F}, \\ \delta_{XZ} & = & \dfrac{\left|\langle X, Z\rangle\right|}{1 + \|X\|_F + \|Z\|_F}, & \delta_{XS} = \dfrac{\|X - \Pi_{\mathcal{N}^n}(X - S)\|}{1 + \|X\|_F + \|S\|_F}. \end{array}$$

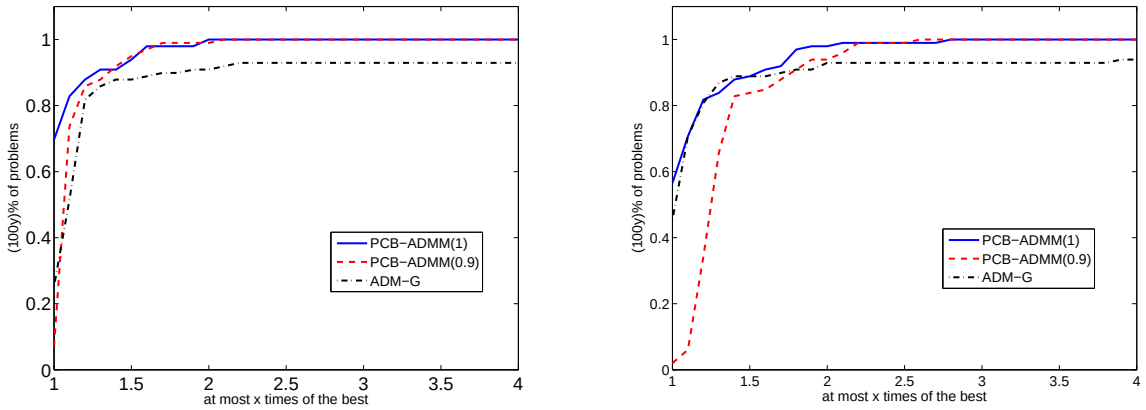
Additionally, we compute the relative gap by

$$\delta_g = \frac{|\text{pobj} - \text{dobj}|}{1 + |\text{pobj}| + |\text{dobj}|},$$

where $\text{pobj} = \frac{1}{2}\langle X, \varphi(X)\rangle + \langle C, X\rangle$ and $\text{dobj} = -\frac{1}{2}\langle X, \varphi(X)\rangle + b^T y$.

We terminate the solvers PCB-ADMM and ADM-G [7] (the step length $\alpha = 0.99$ in the Gaussian back substitution procedure) when $\delta < 10^{-6}$ with the maximum number of iterations set at 25000.

The detailed numerical results are reported in Tables 1-3. Figure 1 shows the performance profiles in terms of the number of iterations and computing time for all the problems tested. We may observe that for the majority of the tested problems, PCB-ADMM with $\alpha = 1$ outperforms PCB-ADMM with $\alpha = 0.9$ and AMD-G in terms of the number of iterations and computing time.



(a) iteration. (b) time.

Figure 1: Performance profiles (iteration and time) of ADM-G and PCB-ADMM (with $\alpha = 1$ and $\alpha = 0.9$ for the problems reported in Table 1-3.

4.2 Low patch-rank decomposition.

The low patch-rank decomposition is to decompose a natural image into two meaningful components. One is the structural, geometrical part and sketchy approximation of image coined as cartoon, and the other one is the oscillations and repeated patterns of image coined as texture. For a given image f, this problem can be mathematically modeled as the following separable 3-block convex programming:

$$\min \qquad \tau_1 \||\nabla u|\|_1 + \tau_2 \|\mathcal{P}v\|_* + \tau_3 \|\mathcal{K}(u+v) - f\|_F^2 \tag{49}$$

where $\||\nabla \cdot |\|_1$ denotes the total variation semi-norm in order to induce the cartoon part u, $\|\cdot\|_*$ denotes the nuclear norm (defined as the sum of all singular values) to reflect the low-rank feature of the matrix $\mathcal{P}v$ where v is the texture part. $\mathcal{P} : \mathbb{R}^n \to \mathbb{R}^{r^2 \times s}$ is a patch mapping to describes the preceding process on rearranging the texture part of an image v as a matrix V, where $s = \lceil \frac{n}{r^2} \rceil$ and $\lceil \cdot \rceil$ is a operator to round a scalar as the nearest integer towards infinity. $\mathcal{K}$ is a linear operator corresponding to certain corruption on the target image f. For more details about $\mathcal{P}$ and $\mathcal{K}$, we refer to [27].

Similar as in [8, 26], we solve the following reformulation of the problem (49):

$$\begin{array}{ll} \min & \theta_1(x_1) + \theta_2(x_2) + \theta_3(x_3) \\ s.t. & \mathcal{A}_1(x_1) + \mathcal{A}_2(x_2) + \mathcal{A}_3(x_3) = 0 \end{array} \tag{50}$$

where

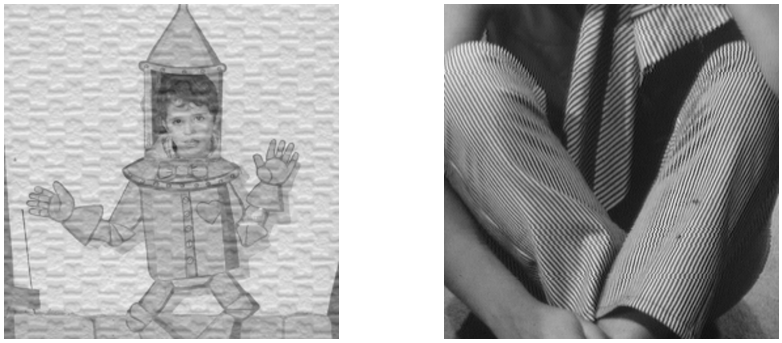
$$\left\{ \begin{array}{l} x_1 = u \\ x_2 = v \\ x_3 = (x, y, z) \end{array} \right. , \quad \left\{ \begin{array}{l} x = \nabla u \\ y = \mathcal{P}v \\ z = u + v \end{array} \right. , \quad \left\{ \begin{array}{l} \theta_1(x_1) = 0 \\ \theta_2(x_2) = 0 \\ \theta_3(x_3) = \tau_1 \||x|\|_1 + \tau_2 \|y\|_* + \frac{\tau_3}{2} \|\mathcal{K}(z) - f\|_F^2. \end{array} \right.$$

and

$$\mathcal{A}_1 = \begin{pmatrix} \nabla \\ 0 \\ \mathcal{I} \end{pmatrix}, \mathcal{A}_2 = \begin{pmatrix} 0 \\ \mathcal{P} \\ \mathcal{I} \end{pmatrix}, \mathcal{A}_3 = \begin{pmatrix} -\mathcal{I} & 0 & 0 \\ 0 & -\mathcal{I} & 0 \\ 0 & 0 & -\mathcal{I} \end{pmatrix}.$$

Since $\mathcal{A}_1^*\mathcal{A}_1$, $\mathcal{A}_2^*\mathcal{A}_2$ and $\mathcal{A}_3^*\mathcal{A}_3$ are invertible, our PCB-ADMM with $\alpha \in (0,1))$ can be applied to solve the problem (50). In testing, we use the BCD order $x_1 \to x_2 \to x_3 \to x_2$ for PCB-ADMM with $\alpha \in (0,1)$. For more detail on solving the subproblems, see [8, 33].

In this section, we test two different cases of (50). One is $\mathcal{K} = S$ in which some pixels of the image f are missing. The other is $\mathcal{K} = B$ in which the image f is corrupted by blur. We report the numerical results obtained by PCB-ADMM, and compare its numerical performance with ADMMe and the partial splitting augmented Lagrangian method (PS-ALD) [26]. The images tested are displayed in Figure 2, the same as that used in [8, 26].



(a) (b)

Figure 2: Test images. (a) 256 × 256 Boy, (b) 256 × 256 Barbara.

The trade-off parameters (τ_1, τ_2, τ_3) in the problem (49) and the scalar r for the patch mapping $\mathcal{P}$ are adopted as suggested in [26, 27]. The parameters for all the solvers (ADMMe, PS-ALD and PCB-ADMM) are set as follows:

(i) For $\mathcal{K} = S$, $\tau_1 = 0.02, \tau_2 = 0.01$ and $\tau_3 = 1$. $r = 11$ for the patch mapping $\mathcal{P}$. $\beta_1 = \beta_2 = 10$ and $\beta_3 = 0.1$.

(ii) For $\mathcal{K} = B$, $\tau_1 = 0.0005, \tau_2 = 0.01$ and $\tau_3 = 1$. $r = 11$ for the patch mapping $\mathcal{P}$. $\beta_1 = \beta_2 = 0.1$, $\beta_3 = 0.01$.

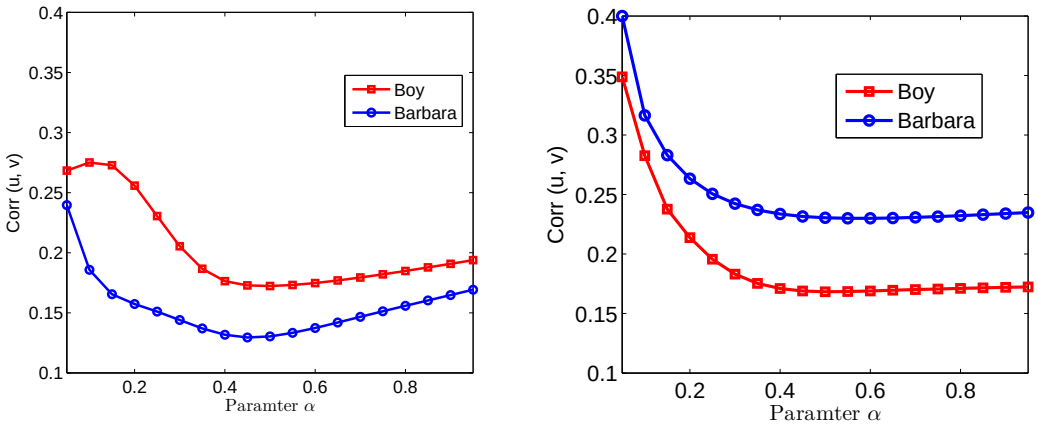
where β_1, β_2 and β_3 are used in the solution of the subproblems [8].

With the correlation of cartoon u and texture v computed by

$$\mathrm{Corr}(u,v) := \frac{\mathrm{cov}(u,v)}{\sqrt{\mathrm{var}(u)\mathrm{var}(v)}},$$

we can measure the quality of image decomposition [32]. Here, $\mathrm{var}(\cdot)$ and $\mathrm{cov}(\cdot,\cdot)$ are the variance and covariance of given variables, respectively. In Figure 3, we plot the evolutions of $\mathrm{Corr}(u,v)$

values with respect to variant α's by PCB-ADMM with $\alpha \in (0, 1)$ for 100 iterations. It shows that the $\alpha \in (0.4, 0.6)$ can yield lower $\mathrm{Corr}(u, v)$ values. Therefore, we set $\alpha = 0.5$ for PCB-ADMM in the following numerical comparisons.



(a) $\mathcal{K} = S$. (b) $\mathcal{K} = B$.

Figure 3: Evolutions of $\mathrm{Corr}(u, v)$ values with respect to variant α's in PCB-ADMM with $\alpha \in [0.05, 0.95]$.

The signal-to-noise ratio (SNR) value is commonly used to measure the quality of the restored or reconstructed images in unit of dB, which is defined as

$$\mathrm{SNR} := 20\log_{10}\frac{\|f^*\|}{\|\bar{f} - f^*\|},$$

where $\bar{f}$ is the approximation of the ground truth f^*. For the images with missing pixels listed in the 1st column of Figure 5, the SNR values are 10.39dB for Boy image and 10.88dB for Barbara image. The blurry images are displayed in the 1st column of Figure 7 with SNRs as 24.18dB for Boy image and 13.18dB for Barbara image.

With the initial iterates as zeros, we run all solvers for 100 iterations and plot the evolutions of objective function values, computing time in seconds and SNRs with respect to iterations in Figure 4 for the case of $\mathcal{K} = S$, and Figure 6 for the case of $\mathcal{K} = B$. The decomposed cartoons and textures for the case of $\mathcal{K} = S$ and $\mathcal{K} = B$ by PCB-ADMM are respectively illustrated in Figure 5 and 7.

We may observe that, ADMMe admits the fastest decay of objective function values, and PCB-ADMM performs comparable or slightly better than PS-ALD. However, the computing time consumed by PCB-ADMM is more than that by ADMMe and PS-ALD, because PCB-ADMM needs to solve the x_2-subproblem twice at each iteration. By PCB-ADMM, the SNR values are 31.03dB and 22.32dB for the reconstructed Boy and Barbara images (4th column in Figure 5) with $\mathcal{K} = S$, and 28.50dB and 15.15dB for the reconstructed Boy and Barbara images (4th column in Figure 7) with $\mathcal{K} = B$.

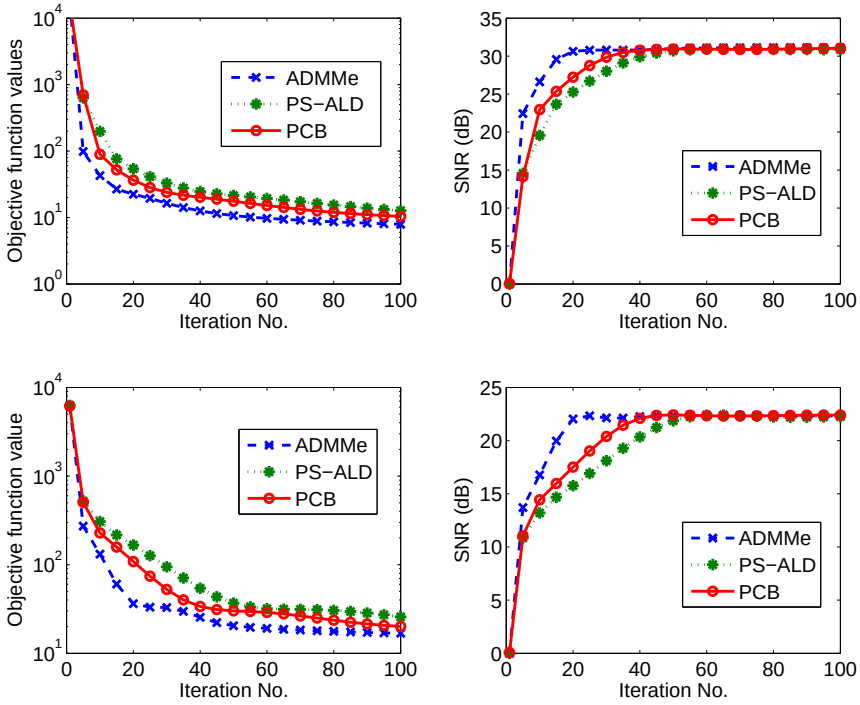


Figure 4: Evolutions of objective function values and SNRs with respect to iterations for the case of $\mathcal{K} = S$, where PCB denotes PCB-ADMM with $\alpha = 0.5$. The first row is for the test image Boy, and the second for Barbara.

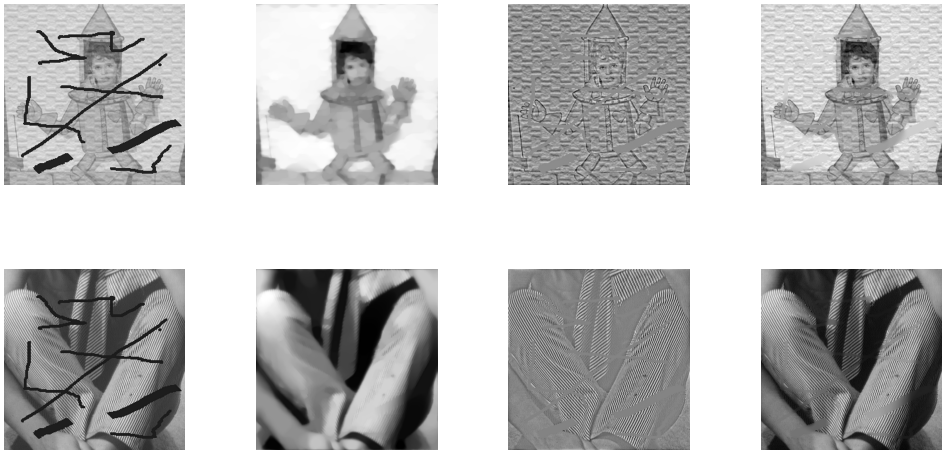


Figure 5: The case of $\mathcal{K} = S$: target images with missing pixels (1st column), decomposed cartoons (2nd column) and textures (3rd column) by PCB-ADMM with $\alpha = 0.5$, reconstructed images (4th column) by superposing the corresponding cartoons and textures

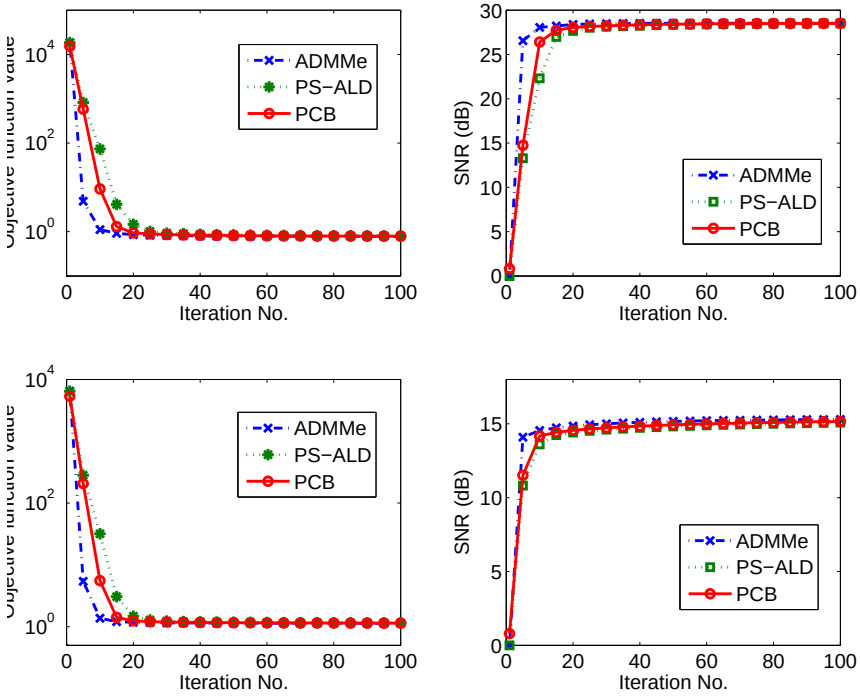


Figure 6: Evolutions of objective function values and SNRs with respect to iterations for the case of $\mathcal{K} = B$, where PCB denotes PCB-ADMM with $\alpha = 0.5$. The first row is for the test image Boy, and the second for Barbara.

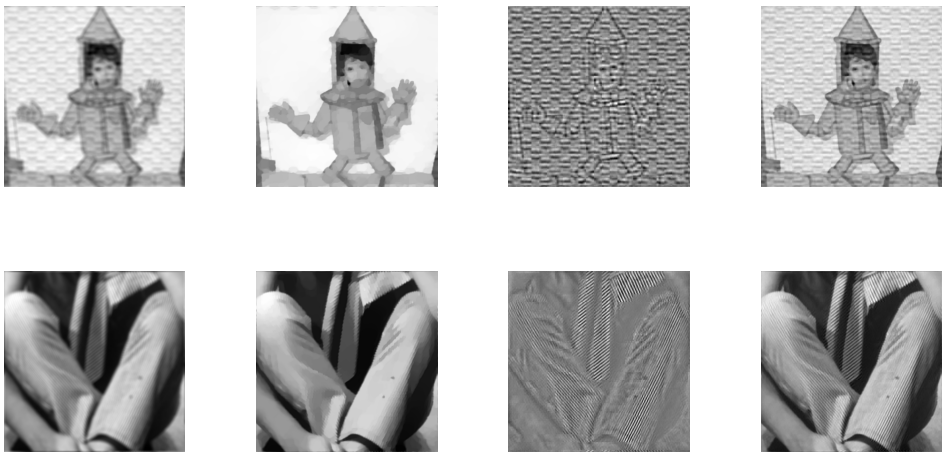


Figure 7: The case of $\mathcal{K} = B$: target images with blurs (1st column), decomposed cartoons (2nd column) and textures (3rd column) by PCB-ADMM with $\alpha = 0.5$, reconstructed images (4th column) by superposing the corresponding cartoons and textures

5 Conclusion.

In this paper, a prediction-correction-based ADMM was presented for solving a special multi-block separable convex programming, with $n-2$ objective functions being linear or quadratic convex. At each iteration, the subproblems are solved by taking a special BCD cycle, and then the outputs are corrected by implementing convex combination of two iteration points from the prediction step and previous iteration.

With the help of the variational inequality, we proved the convergence of the proposed algorithm. The numerical experiments demonstrated that, the prediction-correction-based ADMM is effective and attractive on the convex quadratic semidefinite programming with nonnegative constraints and low patch-rank image decomposition.

References

[1] D.F. Sun, K.C. Toh, L. Yang, A Convergent 3-Block Semi-Proximal Alternating Direction Method of Multipliers for Conic Programming with 4-Type of Constraints, SIAM J. Optim. 25 (2015) 882–915.

[2] X.K. Chang, S.Y. Liu, X. Li, Modified alternating direction method of multipliers for convex quadratic semidefinite programming, Neurocomputing 214 (2016) 575–586.

[3] X.K. Chang, S.Y. Liu, A 2-block semi-proximal ADMM for solving the H-weighted nearest correlation matrix problem, Optimization 66(1) (2017) 1–16.

[4] E.J. Cand'es, X. D. Li, Y. Ma, J. Wright, Robust principal component analysis?, Journal of ACM 58 (2011) 1–37.

[5] T. Bouwmans, N. Aybat, E. Zahzah. Handbook on robust low-rank and sparse matrix decomposition: Applications in image and video processing. CRC Press, Taylor and Francis Group, May 2016.

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

[7] B.S. He, M. Tao, X.M. Yuan, Alternating direction method with gaussian back substitution for separable convex programming, SIAM J. Optim. 22 (2012) 313–340.

[8] B.S. He, X.M. Yuan, A class of ADMM-based algorithms for multi-block separable convex programming, manuscript, 2015.

[9] C. Chen, B.S. He, Y.Y. Ye, X.M. Yuan, The direct extension of admm for multi-block convex minimization problems is not necessarily convergent, Math. Program. Series A (2014) 1–23.

[10] X.D. Li, D.F. Sun, K.-C. Toh, A Schur complement based semi-proximal ADMM for convex quadratic conic programming and extensions. Math. Program. 155(1-2) (2016) 333–373.

[11] L. Chen, D.F. Sun, K.-C. Toh, An efficient inexact symmetric Gauss-Seidel based majorized ADMM for high-dimensional convex composite conic programming, Math. Program. 161(1)(2017) 237–270.

[12] J.J. Wang, W. Song, An algorithm twisted from generalized ADMM for multi-block separable convex minimization models, J. Comput. Appl. Math. 309 (2017) 342–358.

[13] M. Li, D.F. Sun, K.-C. Toh, A Convergent 3-Block Semi-Proximal ADMM for Convex Minimization Problems with One Strongly Convex Block, Asia-Pac. J. of Oper. Res. 32 (04)(2015).

[14] W. Deng and W. Yin. On the global and linear convergence of the generalized alternating direction method of multipliers. Technical report, Rice University CAAM, 2012.

[15] W. Deng, M.J. Lai, Z. Peng, W. Yin, Parallel multi-block ADMM with $O(1/k)$ convergence, J. Sci. Comput. 71(2) (2017) 712–736.

[16] K. Li, M. Sundin, C.R. Rojas, S. Chatterjee, M. Jansson, Alternating strategies with internal ADMM for low-rank matrix reconstruction, Signal Process. 121 (2016) 153–159.

[17] M. Hong, Z.Q. Luo, On the linear convergence of the alternating direction method of multipliers, Math. Program. 162(1-2) (2017) 165–199,

[18] B.S. He, X.M. Yuan, On nonergodic convergence rate of Douglas-Rachford alternating direction method of multipliers, Numerische Mathematik 130 (3) (2015) 567–577.

[19] D.R. Han, X.M. Yuan, Local linear convergence of the alternating direction method of multipliers for quadratic programs, SIAM J. Numer. Anal. 51(6)(2013) 3446–3457.

[20] B.S. He, M. Tao, X.M. Yuan, A splitting method for separable convex programming, IMA J. Numer. Anal. 35(1) (2015) 394–426.

[21] D.R. Han, X.M. Yuan, W.X. Zhang, X.J. Cai,: An ADM-based splitting methods for separable convex programming. Comput. Optim. Appl. 54(2013) 343–369.

[22] R.D.C. Monteiro, B.F. Svaiter, Iteration-complexity of block-decomposition algorithms and the alternating direction method of multipliers. SIAM J. Optim. 23 (2013) 475–507,.

[23] X. Wang, M. Hong, S. Ma, and Z.-Q. Luo, Solving multiple-block separable convex minimization problems using two-block alternating direction method of multipliers, Pac. J. Optim. 11(4) (2015) 645–667.

[24] L. Chen, D.F. Sun, K.-C. Toh, A Note on the Convergence of ADMM for Linearly Constrained Convex Optimization Problems, Comput. Optim. Appl. 66(2)(2017) 327–343.

[25] Rockafellar, R. T.: Convex Analysis. Princeton 1976

[26] D.R. Han, W.W. Kong, W.X. Zhang, A Partial Splitting Augmented Lagrangian Method for Low Patch-Rank Image Decomposition, J. Math. Imaging Vis. 51(2015) 145–160.

[27] H. Schaeffer, S. Osher, A low patch-rank interpretation of texture, SIAM J. Imaging Sci. 6(2013) 226–262.

[28] M.K. Ng, X.M. Yuan, W.X. Zhang, A coupled variational image decomposition and restoration model for blurred cartoon-plus-texture images with missing pixels, IEEE Tran. Imaging Proc., 22(2013) 2233–2246.

[29] M. Mirmehda, X. Xie, S. Suri, Handbook of Texture Analysis. Imperial College Press, London (2008)

[30] K.F. Jiang, D.F. Sun, K.-C. Toh, An inexact accelerated proximal gradient method for large scale linearly constrained convex SDP, SIAM J. Optim. 22 (2012) 1042–1064.

[31] X.K. Chang, S.Y. Liu, P.J. Zhao, A projection method based on KKT conditions for convex quadratic semidefinite programming with nonnegative constraints, http://www.optimization-online.org/DB_HTML/2017/12/6377.html

[32] J. Aujol, G. Gilboa, T. Chan, S. Osher, Structure-texture image decomposition-modeling, algorithms, and parameter selection, Int. J. Comput. Vis. 67(2006) 111–136.

[33] J.F. Yang, X.M. Yuan, Linearized augmented Lagrangian and alternating direction methods for nuclear norm minimization, Math. Comput. 82 (281) (2013) 301–329.

Table 1: The performance of ADM-G and PCB-ADMM (with $\alpha = 1$ and $\alpha = 0.9$) on the problems with $\varphi(X) = X$. In the table, "G", "P(1)" and "P(0.9)" stand for ADM-G, PCB-ADMM with $\alpha = 1$ and PCB-ADMM with $\alpha = 0.9$, respectively.

			iteration	δ	δ_g	time (second)
problem	m_E	n_S	G | P(1) | P(0.9)	G | P(1) | P(0.9)	G | P(1) | P(0.9)	G | P(1) | P(0.9)
theta4	1949	200	553 | 552 | 615	9.9e-7 | 1.0e-6 | 1.0e-6	1.5e-6 | 1.4e-6 | 1.4e-6	5.4 | 5.4 | 6.7
theta42	5986	200	500 | 496 | 521	9.8e-7 | 9.8e-7 | 9.9e-7	7.6e-7 | 7.7e-7 | 8.5e-7	4.5 | 4.6 | 5.4
theta6	4375	300	559 | 549 | 600	9.9e-7 | 9.9e-7 | 9.9e-7	1.1e-6 | 1.1e-6 | 1.1e-6	8.1 | 7.5 | 9.2
theta62	13390	300	440 | 535 | 465	9.9e-7 | 9.9e-7 | 9.8e-7	9.5e-7 | 1.2e-6 | 1.1e-6	6.5 | 7.8 | 7.7
theta8	7905	400	687 | 492 | 539	1.0e-6 | 1.0e-6 | 1.0e-6	9.2e-7 | 7.0e-7 | 9.1e-7	15.8 | 11.3 | 13.6
theta82	23872	400	490 | 559 | 531	9.8e-7 | 1.0e-6 | 9.9e-7	1.4e-6 | 1.4e-6 | 1.4e-6	12.3 | 13.7 | 14.5
theta10	12470	500	617 | 614 | 668	1.0e-6 | 9.9e-7 | 9.9e-7	8.5e-7 | 7.6e-7 | 7.1e-7	23.0 | 21.8 | 26.3
theta102	37467	500	574 | 571 | 616	1.0e-6 | 9.8e-7 | 9.9e-7	1.5e-6 | 1.5e-6 | 1.5e-6	22.1 | 22.0 | 26.3
theta103	62516	500	483 | 482 | 521	9.7e-7 | 9.1e-7 | 9.9e-7	1.8e-6 | 1.7e-6 | 1.8e-6	19.0 | 19.0 | 22.8
theta104	87254	500	512 | 507 | 542	9.8e-7 | 9.8e-7 | 9.9e-7	2.5e-6 | 2.6e-6 | 2.9e-6	20.9 | 20.2 | 23.9
theta123	90020	600	496 | 493 | 534	9.9e-7 | 9.9e-7 | 1.0e-6	2.1e-6 | 2.1e-6 | 1.9e-6	38.7 | 29.3 | 39.3
MANN-a27	703	378	1284 | 1283 | 1443	9.1e-7 | 1.0e-6 | 9.1e-7	2.3e-6 | 2.5e-6 | 2.2e-6	23.0 | 23.1 | 28.8
san200-0.7-1	5971	200	2552 | 2040 | 2192	9.3e-7 | 9.4e-7 | 8.9e-7	2.1e-6 | 2.0e-6 | 2.2e-6	23.5 | 18.7 | 21.7
sanr200-0.7	6033	200	478 | 475 | 503	9.9e-7 | 1.0e-6 | 9.9e-7	9.6e-7 | 9.5e-7 | 8.1e-7	4.4 | 4.3 | 4.9
c-fat200-1	18367	200	1084 | 1069 | 1100	9.9e-7 | 1.0e-6 | 9.9e-7	2.2e-7 | 2.2e-7 | 3.6e-7	9.5 | 9.6 | 10.5
brock200-1	5067	200	489 | 410 | 518	1.0e-6 | 1.0e-6 | 9.8e-7	8.0e-7 | 6.5e-7 | 6.4e-7	4.4 | 3.8 | 5.1
brock200-4	6812	200	409 | 449 | 486	9.9e-7 | 9.8e-7 | 9.8e-7	1.1e-6 | 1.2e-6 | 9.8e-7	3.7 | 4.1 | 4.8
brock400-1	20078	400	571 | 566 | 540	1.0e-6 | 9.9e-7 | 1.0e-6	1.3e-6 | 1.2e-6 | 1.2e-6	13.8 | 14.0 | 14.1
keller4	5101	171	822 | 938 | 876	1.0e-6 | 1.0e-6 | 1.0e-6	4.7e-7 | 5.1e-7 | 4.5e-7	6.1 | 7.0 | 7.1
p-hat300-1	33918	300	1212 | 1201 | 1283	1.0e-6 | 1.0e-6 | 1.0e-6	1.4e-6 | 1.5e-6 | 1.5e-6	18.9 | 18.7 | 21.3
1dc.128	1472	128	1109 | 1054 | 1097	9.9e-7 | 1.0e-6 | 9.9e-7	2.1e-6 | 2.3e-6 | 4.0e-7	7.4 | 7.3 | 8.5
1et.128	673	128	1391 | 1307 | 1307	1.0e-6 | 9.9e-7 | 1.0e-6	1.5e-7 | 1.1e-7 | 1.3e-7	8.4 | 8.1 | 9.0
1tc.128	513	128	1041 | 878 | 921	9.7e-7 | 8.2e-7 | 1.0e-6	2.4e-6 | 4.2e-7 | 1.8e-6	6.5 | 5.3 | 6.2
1zc.128	1128	128	278 | 280 | 303	9.7e-7 | 9.2e-7 | 8.5e-7	2.2e-6 | 2.1e-6 | 1.5e-6	1.6 | 1.6 | 1.9
1dc.256	9728	512	2578 | 2737 | 2876	1.0e-6 | 1.0e-6 | 1.0e-6	4.0e-6 | 4.4e-6 | 4.7e-6	28.7 | 29.4 | 33.1
1et.256	4033	512	2693 | 2690 | 2455	1.0e-6 | 1.0e-6 | 1.0e-6	1.1e-6 | 1.2e-6 | 1.3e-6	29.7 | 29.3 | 28.4
1tc.256	3265	512	5313 | 5207 | 4504	1.0e-6 | 1.0e-6 | 1.0e-6	1.8e-6 | 1.7e-6 | 1.7e-6	55.5 | 57.1 | 51.9
1zc.256	6913	512	295 | 282 | 294	9.7e-7 | 9.4e-7 | 9.8e-7	1.7e-6 | 1.5e-6 | 2.5e-6	3.0 | 2.9 | 3.2
hamming-7-5-6	1793	128	566 | 594 | 678	9.3e-7 | 1.0e-6 | 8.8e-7	1.9e-6 | 1.4e-6 | 1.7e-6	4.0 | 4.2 | 5.5
hamming-8-3-4	16129	256	237 | 228 | 252	9.5e-7 | 7.9e-7 | 9.5e-7	3.5e-7 | 4.0e-7 | 1.4e-7	2.6 | 2.5 | 3.0
hamming-9-5-6	53761	512	521 | 528 | 585	9.5e-7 | 9.8e-7 | 8.4e-7	3.0e-7 | 1.5e-6 | 4.5e-7	26.4 | 24.1 | 31.1
hamming-9-8	2305	512	3153 | 3266 | 3643	9.9e-7 | 9.6e-7 | 9.4e-7	1.6e-6 | 6.3e-7 | 1.6e-6	139.5 | 136.5 | 175.1
hamming-10-2	23041	1024	830 | 845 | 922	8.4e-7 | 9.0e-7 | 1.0e-6	2.6e-6 | 2.8e-6 | 4.0e-6	169.5 | 170.9 | 209.0

Table 2: The performance of ADM-G and PCB-ADMM (with $\alpha = 1$ and $\alpha = 0.9$) on the QSDP problems with $\varphi(X) = \frac{BX+XB}{2}$ ($seed = 1$). In the table, "G", "P(1)" and "P(0.9)" stand for ADM-G, PCB-ADMM with $\alpha = 1$ and PCB-ADMM with $\alpha = 0.9$, respectively.

			iteration	δ	δ_g	time (second)
problem	m_E	n_S	G | P(1) | P(0.9)	G | P(1) | P(0.9)	G | P(1) | P(0.9)	G | P(1) | P(0.9)
theta4	1949	200	432 | 392 | 428	9.9e-7 | 9.9e-7 | 9.9e-7	1.5e-6 | 2.3e-6 | 2.5e-6	5.7 | 5.3 | 6.6
theta42	5986	200	305 | 360 | 400	9.9e-7 | 1.0e-6 | 9.9e-7	1.4e-6 | 1.4e-6 | 1.1e-6	3.5 | 4.4 | 5.8
theta6	4375	300	310 | 468 | 465	9.9e-7 | 9.9e-7 | 9.8e-7	1.3e-6 | 1.8e-6 | 1.8e-6	5.9 | 10.9 | 12.5
theta62	13390	300	274 | 390 | 419	9.9e-7 | 9.6e-7 | 9.9e-7	9.1e-7 | 1.4e-6 | 1.4e-6	5.4 | 9.4 | 11.6
theta8	7905	400	482 | 415 | 452	9.9e-7 | 1.0e-6 | 9.9e-7	1.4e-6 | 1.4e-6 | 1.4e-6	18.8 | 19.9 | 25.0
theta82	23872	400	411 | 444 | 476	9.9e-7 | 1.0e-6 | 9.8e-7	1.7e-6 | 1.9e-6 | 1.9e-6	16.4 | 22.4 | 27.4
theta10	12470	500	756 | 492 | 534	1.0e-6 | 9.7e-7 | 9.9e-7	1.5e-6 | 1.3e-6 | 1.4e-6	50.0 | 39.4 | 51.1
theta102	37467	500	605 | 479 | 479	9.9e-7 | 9.9e-7 | 1.0e-6	1.8e-6 | 2.0e-6 | 1.8e-6	40.6 | 40.0 | 45.8
theta103	62516	500	363 | 459 | 493	9.9e-7 | 8.8e-7 | 9.9e-7	1.6e-6 | 2.3e-6 | 2.3e-6	22.9 | 39.1 | 48.5
theta104	87254	500	382 | 475 | 512	1.0e-6 | 9.9e-7 | 9.9e-7	2.6e-6 | 4.3e-6 | 3.7e-6	24.4 | 40.5 | 50.2
theta123	90020	600	421 | 470 | 507	9.8e-7 | 9.9e-7 | 9.9e-7	2.3e-6 | 3.1e-6 | 3.1e-6	49.9 | 69.6 | 85.5
MANN-a27	703	378	9671 | 921 | 956	1.0e-6 | 9.9e-7 | 9.9e-7	1.5e-6 | 1.2e-6 | 1.3e-6	403.0 | 40.6 | 48.0
san200-0.7-1	5971	200	286 | 443 | 478	9.9e-7 | 9.9e-7 | 1.0e-6	2.3e-6 | 2.3e-6 | 2.3e-6	3.2 | 5.5 | 6.7
sanr200-0.7	6033	200	248 | 389 | 361	9.8e-7 | 9.8e-7 | 1.0e-6	1.1e-6 | 1.4e-6 | 1.5e-6	2.8 | 4.9 | 5.1
c-fat200-1	18367	200	**25000** | 2391 | 2508	**1.5e-6** | 1.0e-6 | 1.0e-6	5.3e-6 | 1.7e-6 | 1.7e-6	303.0 | 30.1 | 35.9
brock200-1	5067	200	276 | 401 | 378	9.9e-7 | 9.9e-7 | 9.9e-7	1.2e-6 | 1.3e-6 | 1.3e-6	3.1 | 4.9 | 5.3
brock200-4	6812	200	251 | 394 | 367	9.7e-7 | 9.9e-7 | 9.9e-7	1.3e-6 | 1.5e-6 | 1.5e-6	2.8 | 4.9 | 5.3
brock400-1	20078	400	431 | 456 | 489	1.0e-6 | 1.0e-6 | 9.9e-7	1.7e-6 | 1.9e-6 | 1.9e-6	17.0 | 23.4 | 27.8
keller4	5101	171	1191 | 638 | 643	1.0e-6 | 9.9e-7 | 9.9e-7	2.3e-7 | 2.4e-7 | 2.5e-7	13.1 | 6.6 | 7.6
p-hat300-1	33918	300	1553 | 809 | 756	1.0e-6 | 1.0e-6 | 1.0e-6	2.2e-6 | 8.8e-7 | 8.6e-7	38.6 | 20.1 | 21.1
1dc.128	1472	128	439 | 640 | 703	9.9e-7 | 9.9e-7 | 9.9e-7	3.4e-6 | 3.1e-6 | 2.5e-6	4.1 | 5.6 | 7.1
1et.128	673	128	12397 | 1218 | 1117	1.0e-6 | 1.0e-6 | 1.0e-6	3.0e-7 | 1.8e-7 | 2.2e-7	107.3 | 9.2 | 10.0
1tc.128	513	128	6822 | 965 | 990	1.0e-6 | 1.0e-6 | 1.0e-6	6.8e-7 | 2.7e-7 | 2.9e-7	56.6 | 7.2 | 8.7
1zc.128	1128	128	254 | 508 | 530	9.9e-7 | 9.9e-7 | 9.8e-7	1.7e-6 | 8.9e-7 | 9.6e-7	1.8 | 3.7 | 4.6
1dc.256	3840	256	1018 | 814 | 900	1.0e-6 | 1.0e-6 | 1.0e-6	9.9e-7 | 1.2e-6 | 1.2e-6	17.4 | 14.3 | 18.4
1et.256	1665	256	18394 | 1296 | 1367	1.0e-6 | 1.0e-6 | 1.0e-6	9.8e-7 | 6.1e-7 | 6.1e-7	368.2 | 22.5 | 27.9
1tc.256	1313	256	15606 | 1872 | 1961	1.0e-6 | 1.0e-6 | 1.0e-6	1.2e-6 | 7.3e-7 | 7.2e-7	276.8 | 32.8 | 39.6
1zc.256	2817	256	1276 | 768 | 795	1.0e-6 | 1.0e-6 | 9.9e-7	1.8e-6 | 9.6e-7 | 1.1e-6	22.6 | 13.4 | 16.0
hamming-7-5-6	1793	128	167 | 320 | 251	8.8e-7 | 9.9e-7 | 9.8e-7	2.3e-6 | 9.8e-8 | 9.9e-8	1.9 | 5.2 | 2.9
hamming-8-3-4	16129	256	**25000** | 5665 | 6021	**4.6e-6** | 1.0e-6 | 1.0e-6	6.6e-6 | 2.0e-6 | 2.1e-6	441.0 | 114.9 | 134.4
hamming-9-5-6	53761	512	**25000** | 11379 | 12019	**4.3e-6** | 1.0e-6 | 1.0e-6	5.1e-5 | 5.0e-6 | 5.0e-6	2001.4 | 1170.9 | 1406.9
hamming-9-8	2305	512	859 | 624 | 672	1.0e-6 | 9.9e-7 | 1.0e-6	1.6e-7 | 2.4e-9 | 4.1e-8	70.2 | 59.6 | 76.7
hamming-10-2	23041	1024	950 | 882 | 926	1.0e-6 | 9.9e-7 | 1.0e-6	6.1e-8 | 1.0e-7 | 1.1e-7	438.9 | 500.0 | 609.3

Table 3: The performance of ADM-G and PCB-ADMM (with $\alpha = 1$ and $\alpha = 0.9$) on the QSDP problems with $\varphi(X) = WXW$ ($seed = 1$). In the table, "G", "P(1)" and "P(0.9)" stand for ADM-G, PCB-ADMM with $\alpha = 1$ and PCB-ADMM with $\alpha = 0.9$, respectively.

			iteration	δ	δ_g	time (second)
problem	m_E	n_S	G | P(1) | P(0.9)	G | P(1) | P(0.9)	G | P(1) | P(0.9)	G | P(1) | P(0.9)
theta4	1949	200	523 | 482 | 506	9.9e-7 | 9.5e-7 | 9.5e-7	3.2e-6 | 3.5e-6 | 4.0e-6	6.7 | 6.3 | 8.2
theta42	5986	200	529 | 483 | 506	9.3e-7 | 9.4e-7 | 1.0e-6	3.0e-6 | 3.8e-6 | 4.6e-6	6.3 | 5.8 | 7.6
theta6	4375	300	497 | 430 | 447	9.5e-7 | 9.8e-7 | 9.7e-7	2.9e-6 | 4.4e-6 | 4.0e-6	10.7 | 9.8 | 12.3
theta62	13390	300	518 | 442 | 456	9.6e-7 | 9.6e-7 | 9.4e-7	3.1e-6 | 4.3e-6 | 4.2e-6	12.2 | 10.3 | 13.1
theta8	7905	400	486 | 424 | 450	9.7e-7 | 1.0e-6 | 9.5e-7	3.0e-6 | 4.2e-6 | 4.0e-6	18.5 | 19.7 | 24.8
theta82	23872	400	515 | 446 | 452	1.0e-6 | 9.7e-7 | 9.6e-7	3.4e-6 | 4.3e-6 | 4.2e-6	20.2 | 21.2 | 25.6
theta10	12470	500	478 | 394 | 423	9.5e-7 | 9.8e-7 | 9.4e-7	3.2e-6 | 4.5e-6 | 4.3e-6	28.7 | 31.7 | 39.6
theta102	37467	500	524 | 444 | 456	9.9e-7 | 9.8e-7 | 9.9e-7	3.3e-6 | 4.2e-6 | 3.2e-6	32.9 | 36.6 | 44.5
theta103	62516	500	528 | 444 | 457	9.8e-7 | 9.4e-7 | 9.7e-7	3.8e-6 | 4.4e-6 | 4.5e-6	34 | 36.5 | 43.7
theta104	87254	500	542 | 456 | 481	9.9e-7 | 9.9e-7 | 9.4e-7	3.8e-6 | 4.7e-6 | 4.5e-6	37.3 | 38.4 | 46.5
theta123	90020	600	543 | 458 | 481	9.7e-7 | 9.8e-7 | 9.6e-7	3.3e-6 | 4.3e-6 | 4.2e-6	58.7 | 66.8 | 79.6
MANN-a27	703	378	495 | 444 | 449	9.5e-7 | 9.7e-7 | 9.6e-7	2.7e-6 | 4.1e-6 | 4.1e-6	16.9 | 19.3 | 22.2
san200-0.7-1	5971	200	528 | 483 | 506	9.8e-7 | 9.4e-7 | 9.9e-7	3.2e-6 | 3.9e-6 | 4.6e-6	7.6 | 5.9 | 7.4
sanr200-0.7	6033	200	529 | 483 | 506	9.4e-7 | 9.4e-7 | 1.0e-6	3.1e-6 | 3.8e-6 | 4.6e-6	6.3 | 6.1 | 7.0
c-fat200-1	18367	200	565 | 491 | 518	9.6e-7 | 9.8e-7 | 9.4e-7	3.8e-6 | 4.5e-6 | 4.3e-6	7.2 | 6.1 | 7.6
brock200-1	5067	200	527 | 483 | 506	9.8e-7 | 9.4e-7 | 1.0e-6	3.2e-6 | 3.8e-6 | 4.5e-6	6.3 | 5.7 | 7.2
brock200-4	6812	200	530 | 483 | 506	9.5e-7 | 9.4e-7 | 9.9e-7	3.1e-6 | 4.0e-6 | 4.7e-6	6.4 | 5.7 | 7.2
brock400-1	20078	400	509 | 441 | 447	1.0e-6 | 9.5e-7 | 9.5e-7	3.3e-6 | 4.2e-6 | 4.2e-6	20.8 | 20.7 | 24.4
keller4	5101	171	522 | 447 | 464	9.6e-7 | 9.6e-7 | 9.7e-7	3.2e-6 | 4.2e-6 | 1.8e-6	5.4 | 4.5 | 6.0
p-hat300-1	33918	300	544 | 467 | 492	9.5e-7 | 9.7e-7 | 9.9e-7	3.7e-6 | 4.6e-6 | 4.7e-6	12.7 | 11.5 | 13.8
1dc.128	1472	128	485 | 434 | 462	9.7e-7 | 9.6e-7 | 9.6e-7	2.2e-6 | 2.7e-6 | 3.0e-6	4.3 | 3.9 | 5.3
1et.128	673	128	471 | 426 | 453	9.5e-7 | 9.7e-7 | 9.8e-7	1.9e-6 | 2.1e-6 | 2.4e-6	3.5 | 3.2 | 4.2
1tc.128	513	128	467 | 424 | 450	9.1e-7 | 9.5e-7 | 1.0e-6	1.8e-6 | 2.0e-6 | 2.4e-6	3.5 | 3.1 | 4.0
1zc.128	1128	128	478 | 430 | 458	9.9e-7 | 1.0e-6 | 9.8e-7	2.0e-6 | 2.5e-6 | 2.7e-6	3.6 | 3.2 | 4.2
1dc.256	3840	256	525 | 459 | 481	9.4e-7 | 9.8e-7 | 9.6e-7	3.0e-6 | 4.3e-6 | 4.8e-6	8.4 | 8.0 | 9.8
1et.256	1665	256	522 | 459 | 480	9.9e-7 | 9.3e-7 | 9.7e-7	3.1e-6 | 4.0e-6 | 4.7e-6	8.2 | 8.2 | 9.4
1tc.256	1313	256	522 | 458 | 480	9.7e-7 | 1.0e-6 | 9.6e-7	3.0e-6 | 4.2e-6 | 4.6e-6	8.4 | 7.9 | 9.5
1zc.256	2817	256	523 | 459 | 480	1.0e-6 | 9.6e-7 | 9.9e-7	3.1e-6 | 4.2e-6 | 4.9e-6	8.3 | 7.9 | 9.6
hamming-7-5-6	1793	128	492 | 441 | 470	9.7e-7 | 9.9e-7 | 9.8e-7	1.9e-6 | 2.4e-6 | 2.7e-6	4.5 | 4.7 | 5.0
hamming-8-3-4	16129	256	525 | 467 | 491	9.3e-7 | 9.9e-7 | 9.6e-7	3.1e-6 | 4.0e-6 | 4.3e-6	9.5 | 10.6 | 10.4
hamming-9-5-6	53761	512	521 | 444 | 459	9.2e-7 | 1.0e-6 | 9.3e-7	3.1e-6 | 4.4e-6 | 4.1e-6	39.5 | 51.3 | 51.8
hamming-9-8	2305	512	497 | 437 | 444	1.0e-6 | 9.5e-7 | 9.8e-7	2.9e-6 | 4.0e-6 | 4.2e-6	34.5 | 48.4 | 48.4
hamming-10-2	23041	1024	554 | 528 | 567	9.4e-7 | 9.8e-7 | 9.8e-7	3.1e-6 | 1.6e-5 | 1.6e-5	200.9 | 320.8 | 374.1