A Partial PPA block-wise ADMM for Multi-Block Constrained Separable Convex Optimization

Yuan Shen[†] [1]
School of Applied Mathematics, Nanjing University of Finance & Economics, Nanjing, 210023, P. R. China.[†]

Abstract: The alternating direction method of multipliers(ADMM) has been proved to be effective for solving two-block separable convex optimization subject to linear constraints. However, it is not necessarily convergent when it is extended to multiple-block case directly. One remedy could be regrouping multiple-block variables into two groups firstly and then adopting the classic ADMM to the resulting model. The two grouped variables are updated in a GaussCSeidel scheme, while the variables within each group are updated in a Jacobi scheme, which could make it very suitable for parallel computing. However, a special proximal term is added to each subproblem in order to derive convergence, which may slow down the convergence. In this paper, we propose a partial PPA ADMM, which only requires the proximal term in the subproblems of the first group, i.e., the subproblems in the second group are intact. At the end of each iteration, we need to do an extension on the variables with fixed step size. As the subproblems in the second group are unchanged, the resulting sequence should yield better quality as well as potentially faster convergence. Compared with several state-of-the-art multi-block ADMM, preliminary numerical experiments on solving both synthetic and real problems show that our proposed method is effective and promising.

Keywords: convex optimization, augmented Lagrangian, alternating direction method of multipliers, multiple-blocks.

AMS classification(2010): 90C30, 65K05, 94A08

1 Introduction

We consider the following grouped multi-block separable convex programming problem

$$\begin{aligned} \min & \sum_{i=1}^{p} f_i(x_i) + \sum_{j=1}^{q} g_j(y_j) \\ \text{s.t.} & \sum_{i=1}^{p} A_i x_i + \sum_{j=1}^{q} B_j y_j = c, \\ & x_i \in \mathcal{X}_i, \quad i = 1, ..., p, \\ & y_j \in \mathcal{Y}_i, \quad i = 1, ..., q. \end{aligned} \tag{1.1}$$

where $f_i(x_i) : \mathcal{R}^{m_i} \to \mathcal{R}$, $g_j(y_j) : \mathcal{R}^{d_j} \to \mathcal{R}$ are closed and proper convex functions (possibly nonsmooth); $A_i \in \mathcal{R}^{n\times m_i}$, $B_j \in \mathcal{R}^{n\times d_j}$ and $c \in\in \mathcal{R}^n$ are given matrices and vectors, respectively; $\mathcal{X}_i \subset \mathcal{R}^{m_i}$ and $\mathcal{Y}_j \subset \mathcal{R}^{d_j}$ are closed convex sets; $p \geq 1$ and $q \geq 1$ are two integers. Throughout this paper, we assume that the solution set of the problem (1.1) is nonempty and all the matrices $A_i, i = 1, ..., p$ and $B_j, j = 1, ..., q$ have full column rank. We denote $A = (A_1, ..., A_p)$, $B = (B_1, ..., B_q)$, $x = (x_1^\top, ..., x_p^\top)^\top$, $y = (y_1^\top, ..., y_q^\top)^\top$, $\mathcal{X} = \mathcal{X}_1 \times ... \times \mathcal{X}_p$, $\mathcal{Y} = \mathcal{Y}_1 \times ... \times \mathcal{Y}_q$ and $\mathcal{M} = \mathcal{X} \times \mathcal{Y} \times \mathcal{R}^n$.

The problem (1.1) has been extensively studied due to its wide applications in various fields, such as the sparse inverse covariance estimation problem [3] in finance and statistics, the compressive sensing problem in signal processing [30], the low rank and sparse representations [24] in image processing and so forth. One

[1] Email: ocsiban@126.com. Research supported by National Natural Science Foundation of China grants 11401295,11726618 and by National Social Science Foundation of China under Grant 17BTQ063 and by Qinglan Project of Jiangsu province.

standard way to solve the problem (1.1) is the classical augmented Lagrangian method (ALM) [21], which minimizes the following augmented Lagrangian function

$$L_\beta(X,Y,\lambda) = L(X,Y,\lambda) + \frac{\beta}{2}\|Ax+By-c\|^2, \tag{1.2}$$

where $\beta > 0$ is a penalty parameter for the equality constraint, and the Lagrange function $L_\beta(X,Y,\lambda)$ of (1.1) is defined by

$$L(X,Y,\lambda) = \sum_{i=1}^{p} f_i(x_i) + \sum_{j=1}^{q} g_j(y_j) - \langle \lambda, Ax+By-c\rangle, \tag{1.3}$$

with $\lambda \in \mathcal{R}^n$ being the Lagrange multiplier associated with constraints $Ax+By=c$. Obviously, $L(X,Y,\lambda)$ and $L_\beta(X,Y,\lambda)$ are both defined on

$$\Omega = X \times Y \times \mathcal{R}^n = X_1 \times ... \times X_p \times Y_1 \times ... \times Y_q \times \mathcal{R}^n. \tag{1.4}$$

The ALM scheme for solving (1.1) can be described as follows

$$\begin{cases} (x^{k+1}, y^{k+1}) = \arg\min\{L_\beta(x,y,\lambda^k) \mid x \in \mathcal{X}, y \in \mathcal{Y}\}, \\ \lambda^{k+1} = \lambda^k - \beta(Ax^{k+1} + By^{k+1} - c). \end{cases} \tag{1.5}$$

However, ALM does not make full use of the separable structure of the objective function of (1.1) and hence, could not take advantage of the special properties of the component objective functions $f_i(x_i)$ and $g_j(y_j)$ in (1.1). As a result, in many recent real applications, especially those involving big data, solving the subproblems of ALM can be very expensive. One effective approach to overcome such difficulty is the alternating direction method of multipliers (ADMM), which was originally proposed in [9] and could be regarded as a splitting version of ALM. At each iteration, ADMM first sequentially optimize over one block variable while fixing all the other block variables, and then follows by updating the Lagrange multiplier. A natural extension of ADMM for solving the multi-block case problem (1.1) takes the following iterations:

$$\begin{cases} x_i^{k+1} = \arg\min\{L_\beta(x_1^{k+1}, ..., x_{i-1}^{k+1}, x_i, x_{i+1}^k, ..., x_p^k, y^k, \lambda^k) \mid x_i \in \mathcal{X}_i\},\ i = 1, ..., p, \\ y_j^{k+1} = \arg\min\{L_\beta(x^{k+1}, y_1^{k+1}, ..., y_{j-1}^{k+1}, y_j, y_{j+1}^k, ..., y_q^k, \lambda^k) \mid y_j \in \mathcal{Y}_j\},\ j = 1, ..., q, \\ \lambda^{k+1} = \lambda^k - \beta(Ax^{k+1} + By^{k+1} - c). \end{cases} \tag{1.6}$$

Obviously, the scheme (1.6) is a serial algorithm which uses the newest information of the variables at each iteration, and it was proved to be globally convergent for the two-block case ($p = q = 1$). The effectiveness of this directly extended ADMM is numerically verified by solving three-block separable models arising from robust principal component analysis (RPCA), see, e.g., [26, 25]. In the recent works [10, 15, 22, 8], the authors established the convergence of (1.6) under some additional assumptions, such as strongly convexity of all $f_i(x_i)$. However, a counter example was given in [7] to show that the extended ADMM does not necessarily converge for the multi-block case ($p + q \geq 3$) without modifications or additional assumptions.

To ensure the convergence of multi-block ADMM, one way is to correct the output of (1.6) via a simple correction step, then some ADMM-based algorithms with convergence guarantee can be dereived [17, 18, 13]. The other way is to modify the scheme (1.6) by either modifying the subproblem, e.g., see [19, 23, 29], or changing the updating order, e.g., see [6].

A natural extension of ADMM is to use the Jacobian fashion, where the variables are updated simultaneously after each iteration, that is,

$$\begin{cases} x_i^{k+1} = \arg\min\{L_\beta(x_1^k, ..., x_{i-1}^k, x_i, x_{i+1}^k, ..., x_p^k, y^k, \lambda^k) \mid x_i \in \mathcal{X}_i\},\ i = 1, ..., p, \\ y_j^{k+1} = \arg\min\{L_\beta(x^k, y_1^k, ..., y_{j-1}^k, y_j, y_{j+1}^k, ..., y_q^k, \lambda^k) \mid y_j \in \mathcal{Y}_j\},\ j = 1, ..., q, \\ \lambda^{k+1} = \lambda^k - \beta(Ax^{k+1} + By^{k+1} - c). \end{cases} \tag{1.7}$$

It is intuitively easy to understand and also well justified for solving, say, linear equations that GaussCSeidel-type splitting methods which incorporates the latest information as soon as possible generally have faster

convergence than the corresponding Jacobian-type splitting methods. However, the Jacobian splitting method (1.7) enjoys a simultaneous procedure for solving the $p+q$ subproblems when parallel computing infrastructures are available, which becomes increasingly important for solving high-dimensional problems, hence it often outperforms the former by taking less computing time, especially for large-scale problems, due to its fitness to parallel computing, see, e.g., [11, 13, 16].

As shown in [16], however, the Jacobian scheme (1.7) is not necessarily convergent either. To guarantee global convergence, certain variants of (1.7) in the prediction-correction fashion are proposed in [14, 13, 16], which update the output of (1.7) with a further correction step, similar to the variants of (1.6).

However, these correction steps at each iteration could be costly. We note that although incorporating an extra correction step is a popular strategy of modifying the GaussCSeidel and the Jacobian type splitting methods so that global convergence can be guaranteed for multi-block case, such an extra correction step not only requires more storage space for the temporary outputs of (1.6) and (1.7) but also introduces a significant computation load for many applications. Therefore, how to simplify or even remove the potentially expensive correction step is very important in improving the numerical performance of relevant splitting algorithms.

Based on this idea, He et al. [19] proposed a ADMM-type splitting method (entitled HTY method) by adding certain proximal terms which allows some of the subproblems to be solved in parallel, i.e., in a Jacobian fashion. The resulting algorithm is as follows:

$$\begin{cases} x^{k+1} = \arg\min\{L_\beta(x, y_1^k, ..., y_q^k, \lambda^k) \mid x \in \mathcal{X}\}, \\ y_j^{k+1} = \arg\min\{L_\beta(x^{k+1}, y_1^k, ..., y_{j-1}^k, y_j, y_j^k, ..., y_q^k, \lambda^k) + \frac{\tau\beta}{2}\|B_j(y_j - y_j^k\|^2 \mid y_j \in \mathcal{Y}_j\}, \ j = 1, ..., q, \\ \lambda^{k+1} = \lambda^k - \beta(Ax^{k+1} + By^{k+1} - c), \end{cases} \tag{1.8}$$

where $\tau > q - 1$, and a proximal term in the form of, say $\frac{\tau\beta}{2}\|B_j(y_i - y_i^k\|^2$, was added to each subproblem to ensure the global convergence. Actually, HTY method is in fact a mixture of Jacobi and Gauss-Seidel schemes which enjoys the same good feature as the ADMM-related method, and extensive numerical results further verified the efficiency of the proposed method.

Hou et al. [23] improved the HTY method by incorporating the customized PPA technique, and proposed a partial parallel splitting method in which the updating of multipliers is done before computing the rest $q-1$ subproblems. This method can be interpreted as a special application of PPA to the original problem, hence the updated variables at each iteration can be extended by a factor close to 2. Wang and Song proposed a twisted version of ADMM [27] which is similar to that in [23], where it allows the proximal matrix to be indefinite such that the numerical performance can be even better.

As can be seen (1.8), the HTY method can not fully exploit the separability of the objective, i.e., we have to compute the first subproblem before computing the rest subproblems. As a remedy, He and Yuan [4] proposed an entitled block-wise ADMM as follows:

$$\begin{cases} x_i^{k+1} = \arg\min\{L_\beta(x_1^k, ..., x_{i-1}^k, x_i, x_{i+1}^k, ..., x_p^k, y^k, \lambda^k) + \frac{\tau_1\beta}{2}\|A_i(x_i - x_i^k\|^2 \mid x_i \in \mathcal{X}_i\}, \ i = 1, ..., p, \\ y_j^{k+1} = \arg\min\{L_\beta(x^{k+1}, y_1^k, ..., y_{j-1}^k, y_j, y_j^k, ..., y_q^k, \lambda^k) + \frac{\tau_2\beta}{2}\|B_j(y_j - y_j^k\|^2 \mid y_j \in \mathcal{Y}_j\}, \ j = 1, ..., q, \\ \lambda^{k+1} = \lambda^k - \beta(Ax^{k+1} + By^{k+1} - c). \end{cases} \tag{1.9}$$

where a proximal term similar to that in (1.8) was also added to each subproblem to ensure the global convergence with $\tau_1 > p, \tau_2 > q$. The numerical efficiency of block-wise ADMM was further improved by introducing a relaxation factor on the λ update, see [20]. At the cost of parallel computing, this method requires all subproblem to include proximal terms.

Based on (1.9), Bai et al. [2] proposed a generalized symmetric ADMM (GS-ADMM), in which the multipliers are updated two times in each iteration, and the condition on dual step sizes are relaxed. The GS-ADMM also requires all subproblem to include proximal terms, but it is reported to outperform several efficient ADMM-based algorithms including HTY method. Very recently, Bai and Zhang [1] also proposed a Middle Proximal ADMM, in which the first and the last subproblems do not need to add proximal terms, i.e., two subproblems are intact.

In this paper, we study along this line and propose a new partially parallel splitting ADMM method, in which at most three subproblems can be intact, thus it potentially has better numerical behaviour than the existing method. The rest of this paper is organized as follows. In Sect. 2, we define our notation and give some preliminaries for the subsequent analysis. In Sect. 3, we formally present the proposed algorithm and

establish all the convergence results. In Sect. 4, extensive numerical experiments will be conducted on both synthetic and real data to compare our method with some state-of-the-art existing methods. Finally, we give some concluding remarks in Sect. 5.

2 Preliminaries

In this section, we first summarize some notation that will be used throughout this paper. We then use a variational inequality to characterize the optimality conditions of (1.1) and recall some well-known results that will play central roles in the later analysis.

2.1 A variational inequality characterization

We denote by $x^* = (x_1^*, ..., x_p^*)$ and $y^* = (y_1^*, ..., y_q^*)$. Let (x^*, y^*, λ^*) be a saddle point of the Lagrange function (1.1). Then, for any $\lambda \in \mathcal{R}^n$, $x \in X$, $y \in Y$, we have

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

Indeed, finding a saddle point of $L(x, y, \lambda)$ can be expressed as the following variational inequalities: $(x_1^*, ..., x_p^*, y_1^*, ..., y_q^*, \lambda^*) \in \Omega$ where Ω is defined in (1.4), such that

$$\begin{cases} x_i^* \in \mathcal{X}_i,\ f_i(x_i) - f_i(x_i^*) + (x_i - x_i^*)^\top(-A_i^\top \lambda^*) \geq 0,\ \forall x_i \in \mathcal{X}_i,\ i = 1, ..., p, \\ y_j^* \in \mathcal{Y}_j,\ g_j(y_j) - g_j(y_j^*) + (y_j - y_j^*)^\top(-B_j^\top \lambda^*) \geq 0,\ \forall y_j \in \mathcal{Y}_j,\ j = 1, ..., q, \\ \lambda^* \in \mathcal{R}^n, \quad (\lambda - \lambda^*)^\top(Ax^* + By^* - c),\ \forall \lambda \in \mathcal{R}^n. \end{cases} \tag{2.1}$$

More compactly, the variational inequalities in (2.1) can be rewritten in a compact form

$$VI(\Omega, F, f) \quad w^* \in \Omega, \quad f(u) - f(u^*) + (w - w^*)^\top F(w^*) \geq 0, \quad \forall w \in \Omega, \tag{2.2}$$

with

$$f(x) = \sum_{i=1}^{p} f_i(x_i), \quad g(y) = \sum_{j=1}^{q} g_j(y_j), \quad \theta(u) = f(x) + g(y), \tag{2.3}$$

$$u = \begin{pmatrix} x_1 \\ \vdots \\ x_p \\ y_1 \\ \vdots \\ y_q \end{pmatrix}, w = \begin{pmatrix} x_1 \\ \vdots \\ x_p \\ y_1 \\ \vdots \\ y_q \\ \lambda \end{pmatrix}, F(w) = \begin{pmatrix} -A_1^\top \lambda \\ \vdots \\ -A_p^\top \lambda \\ -B_1^\top \lambda \\ \vdots \\ -B_q^\top \lambda \\ Ax + By - c \end{pmatrix}. \tag{2.4}$$

In what follows, we denote (2.2) by VI(Ω,F,f). It is easy to verify that $(\tilde{w} - \hat{w})^\top (F(\tilde{w}) - F(\hat{w})) = 0$ for any $\tilde{w}, \hat{w} \in \Omega$, and thus the mapping $F(\cdot)$ defined in (2.4) is monotone. We also denote by Ω^* the set of all saddle points of $L(x, y, \lambda)$, and through out this paper, it is always assumed that the solutions set of VI(Ω,F,f), denoted by W^* is nonempty. This variational inequality characterizes the first-order optimality condition of the block-wise original model (1.1). We need this variational inequality characterization for the upcoming theoretical analysis.

We then show some useful results for our analysis.

Lemma 2.1 *For the matrices A and B defined in Section 1, we have the following conclusions:*

$$p \cdot diag(A^\top A) \succeq A^\top A \quad and \quad q \cdot diag(B^\top B) \succeq B^\top B, \tag{2.5}$$

where $diag(A^\top A)$ and $diag(B^\top B)$ are defined by

$$diag(A^\top A) = \begin{pmatrix} A_1^\top A_1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & A_p^\top A_p \end{pmatrix} \quad and \quad diag(B^\top B) = \begin{pmatrix} B_1^\top B_1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & B_q^\top B_q \end{pmatrix}, \tag{2.6}$$

respectively.

The proof is omitted as it is trivial. Using the above lemma, it is easy to verify that

$$\tau \mathrm{diag}(A^\top A) - A^\top A \succ 0 \quad \text{with} \quad \tau > p \quad \text{and} \quad \tau \mathrm{diag}(B^\top B) - B^\top B \succ 0 \quad \text{with} \quad \tau > q. \tag{2.7}$$

3 A new partial PPA block-wise ADMM

In this section, we first describe our new partial PPA block-wise ADMM and then establish its convergence properties in the framework of variational inequality.

3.1 Algorithm

Our new algorithm is as follows:

Algorithm 1: A partial PPA block-wise ADMM for (1.1).
Initialization: Regroup the primal variables into two groups ($q \leq 3$), set$\tau > p - 1$, $\alpha \in (0, 2 - \sqrt{q})$.
With given $w^k = (x^k, y^k, \lambda^k)$, the new iterate w^{k+1} is generated by the following steps.

$$\begin{cases} \bar{x}_i^k = \arg\min\{L_\beta(x_1^k, ..., x_{i-1}^k, x_i, x_{i+1}^k, ..., x_p^k, y^k, \lambda^k) + \frac{\tau\beta}{2}\|A_i(x_i - x_i^k)\|^2 \mid x_i \in \mathcal{X}_i\},\ i = 1, ..., p, \\ \bar{y}_j^k = \arg\min\{L_\beta(\bar{x}^k, y_1^k, ..., y_{j-1}^k, y_j, y_j^k, ..., y_q^k, \lambda^k) \mid y_j \in \mathcal{Y}_j\},\ j = 1, ..., q, \\ \bar{\lambda}^k = \lambda^k - \beta(A\bar{x}^k + B\bar{y}^k - c), \\ w^{k+1} = w^k - \alpha(w^k - \bar{w}^k). \end{cases} \tag{3.1}$$

Remark 3.1 *Compared with the block-wise ADMM (1.9), the most prominent feature of the scheme (3.1) is that the subproblem in the second group is intact, which could imply the better "quality" of the resulting sequence. It is also noticeable that an extension step is applied at the end of each iteration to guarantee the convergence, and the setting of extension step size α will be elaborated latter. Note the proximal terms aims at controlling the proximity of the solutions of the decomposed subproblems, and we do not apply the proximal terms to subproblems in the second group, hence the scheme (3.1) is more likely to converge when the number of blocks in the second group, i.e., q, is small. To this end, we restrict the maximal setting of q to be 3.*

We do a simple comparison between the proposed algorithm and some related algorithms:

- Compared with the block-wise ADMM [4] which requires proximal terms in all subproblems, our algorithm only requires proximal terms in the first group of subproblems. At the cost of less subproblems with proximal terms, there is an additional extension step on all variables at the end of each iteration. In addition, our algorithm has less freedom in choosing grouping strategy due to limit of the number of blocks in the second group.

- The splitting method in [19] (HTY method) does not require proximal terms in the first subproblem, but its grouping strategy is fixed (1∼(m-1)). In contrary, our algorithm can adapt to at most 3 grouping strategies which makes our algorithm more flexibility.

- The partial splitting augmented Lagrangian (PSAL) method [12] does not require proximal terms in all subproblems, but it is only designed for 3-block case. In 3-block case, it is very similar to our algorithm with 1∼2 grouping strategy, and the only difference lies in the extension step. In PSAL method, the extension is done on x_3 and λ with extension factor not greater $(3-\sqrt{5})/2 \approx 0.38$, while in our algorithm, the extension is done on all variables with extension factor not greater $2 - \sqrt{2} \approx 0.58$.

- In the recently proposed a middle proximal ADMM (MPADMM) [1], two subproblems (first and last) are allowed to be free of proximal term, and the extension is done on the middle part of variables with an extension factor which is related to the proximal parameter.

3.2 Convergence analysis

In this section, we analyze the convergence for the proposed partial PPA block-wise ADMM (3.1). We will prove the global convergence, and establish the worst-case convergence rate measured by both the ergodic and a nonergodic senses.

In the following analysis, we need to introduce the auxiliary variables as follows:

$$\tilde{x}^k = \bar{x}^k, \quad \tilde{y}^k = \bar{y}^k, \quad \text{and} \quad \tilde{\lambda}^k = \lambda^k - \beta(A\bar{x}^k + By^k - c), \tag{3.2}$$

Recall the notation $w = (x, y, \lambda) = (x_1, ..., x_p, y_1, .., y_q, \lambda)$ with superscripts; and further define $\tilde{\lambda} = (\tilde{x}, \tilde{y}, \tilde{\lambda})$ with superscripts, we can rewrite the iterative scheme of (3.1) as the following prediction-correction framework which is more beneficial for our analysis.

Prediction:

$$\begin{aligned}
\tilde{x}_i^k &= \arg\min\{L_\beta(x_1^k, ..., x_{i-1}^k, x_i, x_{i+1}^k, ..., x_p^k, y^k, \lambda^k) + \frac{\tau\beta}{2}\|A_i(x_i - x_i^k)\|^2 \mid x_i \in \mathcal{X}_i\},\ i = 1, ..., p, \\
\tilde{y}_j^k &= \arg\min\{L_\beta(\tilde{x}^k, y_1^k, ..., y_{j-1}^k, y_j, y_j^k, ..., y_q^k, \lambda^k) \mid y_j \in \mathcal{Y}_j\},\ j = 1, ..., q, \\
\tilde{\lambda}^k &= \lambda^k - \beta(A\tilde{x}^k + By^k - c),
\end{aligned} \tag{3.3}$$

Correction:

$$w^{k+1} = w^k - \alpha M(w^k - \tilde{w}^k), \quad \text{where} \quad M = \begin{pmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & -\beta B & I \end{pmatrix}. \tag{3.4}$$

The reason for applying an extension step with step size α is to produce a strictly contractive sequence with respect to the solution set of (1.1). In addition, it turns out that the progress of proximity to the solution set at each iteration can be measured by the quantity $\|w^k - \tilde{w}^k\|$.

To prove the global convergence of the scheme (3.1), we derive the following basic lemma.

Lemma 3.1 *For the iterates $\tilde{w}^k$ defined in (3.2), we have*

$$\tilde{w}^k \in \Omega, \quad \theta(u) - \theta(\tilde{u}^k) + (w - \tilde{w}^k)^\top F(\tilde{w}^k) \geq (w - \tilde{w}^k)^\top Q(w^k - \tilde{w}^k), \quad \forall w \in \Omega, \tag{3.5}$$

where

$$Q = \begin{pmatrix} (\tau+1)\beta diag(A^\top A) - \beta A^\top A & 0 & 0 \\ 0 & \beta diag(B^\top B) & 0 \\ 0 & -B & \frac{1}{\beta} I \end{pmatrix}. \tag{3.6}$$

Proof: First, the x_i-subproblem in (3.3) can be written as

$$\begin{aligned}
\tilde{x}_i^k &= \arg\min\{L_\beta(x_1^k, ..., x_{i-1}^k, x_i, x_{i+1}^k, ..., x_p^k, y^k, \lambda^k) + \frac{\tau\beta}{2}\|A_i(x_i - x_i^k\|^2 \mid x_i \in \mathcal{X}_i\},\ i = 1, ..., p, \\
&= \arg\min\{f_i(x_i) - \langle \lambda^k, A_i x_i\rangle + \frac{\beta}{2}\|A_i(x_i - x_i^k) + Ax^k + By^k - c\|^2 + \frac{\tau\beta}{2}\|A_i(x_i - x_i^k\|^2 \mid x_i \in \mathcal{X}_i\},
\end{aligned} \tag{3.7}$$

where some constant terms are ignored in its objective function. The first-order optimality condition of the above convex minimization problem can be written as

$$\begin{aligned}
\tilde{x}_i^k \in \mathcal{X}_i \quad & f_i(x_i) - f_i(\tilde{x}_i^k) + (x_i - \tilde{x}_i^k)^\top \{-A_i^\top \lambda^k \\
& + \beta A_i^\top [A_i(\tilde{x}_i^k - x_i^k) + Ax^k + By^k - c] + \tau\beta A_i^\top A_i(\tilde{x}_i^k - x_i^k)\} \geq 0,\ \forall x_i \in \mathcal{X}_i.
\end{aligned} \tag{3.8}$$

Plugging $\lambda^k = \tilde{\lambda}^k + \beta(A\tilde{x}^k + By^k - c)$ (see (3.3)) into (3.8), we obtain

$$\begin{aligned}
\tilde{x}_i^k \in \mathcal{X}_i \quad & f_i(x_i) - f_i(\tilde{x}_i^k) + (x_i - \tilde{x}_i^k)^\top \{-A_i^\top \tilde{\lambda}^k \\
& - \beta A_i^\top A(\tilde{x}_i^k - x_i^k) + (\tau+1)\beta A_i^\top A_i(\tilde{x}_i^k - x_i^k)\} \geq 0,\ \forall x_i \in \mathcal{X}_i.
\end{aligned} \tag{3.9}$$

Taking $i = 1, ..., p$ in the above variational inequality and grouping them, we obtain that $\tilde{x}^k \in \mathcal{X}$ and

$$\begin{aligned}
\tilde{x}^k \in \mathcal{X} \quad & f(x) - f(\tilde{x}^k) + (x - \tilde{x}^k)^\top \{-A^\top \tilde{\lambda}^k \\
& - \beta A^\top A(\tilde{x}^k - x^k) + (\tau+1)\beta \mathrm{diag}(A^\top A)(\tilde{x}^k - x^k)\} \geq 0,\ \forall x \in \mathcal{X}.
\end{aligned} \tag{3.10}$$

Following the above procedure, we can also obtain that $\tilde{y}^k \in \mathcal{Y}$ and:

$$g(y) - g(\tilde{y}^k) + (y - \tilde{y}^k)^\top \{-B^\top \tilde{\lambda}^k + \beta \mathrm{diag}(B^\top B)(\tilde{y}^k - y^k)\} \geq 0, \ \forall y \in \mathcal{Y}. \tag{3.11}$$

Recalling the definition of $\tilde{\lambda}^k$ in (3.3) again, we get

$$(A\tilde{x}^k + B\tilde{y}^k - c) - B(\tilde{y}^k - y^k) + \frac{1}{\beta}(\tilde{\lambda}^k - \lambda^k) = 0, \tag{3.12}$$

or equivalently

$$\tilde{\lambda}^k \in \mathcal{R}^n, \quad (\lambda - \tilde{\lambda}^k)^\top \{(A\tilde{x}^k + B\tilde{y}^k - c) - B(\tilde{y}^k - y^k) + \frac{1}{\beta}(\tilde{\lambda}^k - \lambda^k)\} \geq 0, \quad \forall \lambda \in \mathcal{R}^n. \tag{3.13}$$

Combining (3.10), (3.11) and (3.13) together and using the notations $F(w)$ and Q (see (2.4) and (3.6)), the assertion of this theorem is followed directly.

Theorem 3.1 *For given matrices Q in (3.6) and M in (3.4). Assuming $q \leq 3$ and setting $\alpha \in (0, 2 - \sqrt{q})$. Let*

$$H = QM^{-1} \quad \textit{and} \quad G = Q^\top + Q - \alpha M^\top H M, \tag{3.14}$$

then, both H and G are positive definite.

Proof: First, we show the positive definiteness of the matrix H. For the matrix M defined in (3.4), we have

$$M^{-1} = \begin{pmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & \beta B & I \end{pmatrix}. \tag{3.15}$$

According to the definition of the matrix H (see (3.14)), we have

$$\begin{aligned} H &= QM^{-1} && (3.16)\\ &= \begin{pmatrix} (\tau+1)\beta \mathrm{diag}(A^\top A) - \beta A^\top A & 0 & 0 \\ 0 & \beta \mathrm{diag}(B^\top B) & 0 \\ 0 & -B & \frac{1}{\beta} I \end{pmatrix} \begin{pmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & \beta B & I \end{pmatrix} && (3.17)\\ &= \begin{pmatrix} (\tau+1)\beta \mathrm{diag}(A^\top A) - \beta A^\top A & 0 & 0 \\ 0 & \beta \mathrm{diag}(B^\top B) & 0 \\ 0 & 0 & \frac{1}{\beta} I \end{pmatrix} \end{aligned}$$

The positive definiteness of H follows from (2.7) and the assumption on B directly. Now, we turn to prove that the matrix G is positive definite. First, we have the identity

$$\begin{aligned} G &= Q^\top + Q - \alpha M^\top H M && (3.18)\\ &= M^\top H + H M - \alpha M^\top H M && (3.19)\\ &= \begin{pmatrix} G_{11} & 0 \\ 0 & G_{22} \end{pmatrix}, \end{aligned}$$

where

$$G_{11} = (2-\alpha)\{(\tau+1)\beta \mathrm{diag}(A^\top A) - \beta A^\top A\}, G_{22} = \begin{pmatrix} (2-\alpha)\beta \mathrm{diag}(B^\top B) - \alpha\beta B^\top B & -(1-\alpha)B^\top \\ -(1-\alpha)B & \frac{2-\alpha}{\beta} I \end{pmatrix}. \tag{3.20}$$

Recalling (2.7) and noting $\alpha \in [0, 1]$, G_{11} is always positive definite, and the remaining task is to verify the positive definiteness of G_{22}.

In fact, G_{22} can be factorized into the following form:

$$G_{22} = \begin{pmatrix} \sqrt{\beta} B_1^\top \\ \vdots \\ \sqrt{\beta} B_q^\top \\ \frac{1}{\sqrt{\beta}} I \end{pmatrix} \bar{G}_{22} \begin{pmatrix} \sqrt{\beta} B_1 & \cdots & \sqrt{\beta} B_q & \frac{1}{\sqrt{\beta}} I \end{pmatrix}, \tag{3.21}$$

where

$$\bar{G}_{22} = \begin{pmatrix} (2-2\alpha)I & -\alpha I & \cdots & -\alpha I & -(1-\alpha)I \\ -\alpha I & (2-2\alpha)I & & & \vdots \\ \vdots & & \ddots & -\alpha I & \\ -\alpha I & & -\alpha I & (2-2\alpha)I & -(1-\alpha)I \\ -(1-\alpha)I & \cdots & & -(1-\alpha)I & (2-\alpha)I \end{pmatrix}. \tag{3.22}$$

Based on our assumptions, the positive definiteness of G_{22} is equivalent to that of $\bar{G}_{22}$, and is further equivalent to that of the following matrix:

$$\tilde{G}_{22} = \begin{pmatrix} 2-2\alpha & -\alpha & \cdots & -\alpha & -(1-\alpha) \\ -\alpha & 2-2\alpha & & & \vdots \\ \vdots & & \ddots & -\alpha & \\ -\alpha & & -\alpha & 2-2\alpha & -(1-\alpha) \\ -(1-\alpha) & \cdots & & -(1-\alpha) & 2-\alpha \end{pmatrix}. \tag{3.23}$$

When q is small, we can calculate all the sequential principal minor determinants of the above matrix by hand. For instance, when $q = 1$, the principal minor determinants of (3.23) are $2-2\alpha$ and $(2-2\alpha)(2-\alpha)-(1-\alpha)^2$. We then let them to be greater than 0, and derive the feasible range of α. By simple calculation, it can be verified that $\alpha \in (0, 2-\sqrt{q}) \Leftrightarrow \tilde{G}_{22} \succ 0$. The assertion is proved.

Remark 3.2 *According to Theorem 3.1, to guarantee the positive definiteness of matrix G, the number of blocks in the second group (i.e., q) can not be greater than 3, and q also determines the range of extension factor α: the larger q is, the smaller α should be. In fact, larger q means less subproblems be controlled by proximal term, hence we should use smaller α to guarantee the convergence.*

Based on the conclusions in the above two theorems, we are able to analyze the convergence for the scheme (3.1). In the following theorems, we prove the global convergence for the scheme (3.1).

Theorem 3.2 *Let $\{w^k\}$ be the sequence generated by the scheme (3.1), and $\tilde{w}^k$, H, M and G are defined in (3.3), (3.4) and (3.14), respectively. Then we have*

$$\theta(u) - \theta(\tilde{u}^k) + (w - \tilde{w}^k)^\top F(\tilde{w}^k) \geq \frac{1}{2\alpha}(\|w - w^{k+1}\|_H^2 - \|w - w^k\|_H^2) + \frac{1}{2}\|w^k - \tilde{w}^k\|_G^2, \quad \forall w \in \Omega. \tag{3.24}$$

Proof: Recalling $Q = HM$ and the relation (3.4), the right-hand side of (3.5) can be written as

$$\frac{1}{\alpha}(w - \tilde{w}^k)^\top H(w^k - w^{k+1}), \tag{3.25}$$

then we have

$$\theta(u) - \theta(\tilde{u}^k) + (w - \tilde{w}^k)^\top F(\tilde{w}^k) \geq \frac{1}{\alpha}(w - \tilde{w}^k)^\top H(w^k - w^{k+1}), \quad \forall w \in \Omega. \tag{3.26}$$

Applying the identity

$$(a-b)^\top H(c-d) = \frac{1}{2}(\|a-d\|_H^2 - \|a-c\|_H^2) - \frac{1}{2}(\|c-b\|_H^2 - \|d-b\|_H^2) \tag{3.27}$$

to the right-hand side of (3.26) with

$$a = w, \quad b = \tilde{w}^k, \quad c = w^k, \quad \text{and} \quad d = w^{k+1}, \tag{3.28}$$

we get

$$(w - \tilde{w}^k)^\top H(w^k - w^{k+1}) = \frac{1}{2}(\|w - w^{k+1}\|_H^2 - \|w - w^k\|_H^2) - \frac{1}{2}(\|w^k - \tilde{w}^k\|_H^2 - \|w^{k+1} - \tilde{w}^k\|_H^2). \tag{3.29}$$

For the term in the last round bracket of the right hand side of (3.29), we have

$$\begin{aligned}\|w^k-\tilde{w}^k\|_H^2 &- \|w^{k+1}-\tilde{w}^k\|_H^2 \\ &= \|w^k-\tilde{w}^k\|_H^2-\|(w^k-\tilde{w}^k)-(w^k-w^{k+1})\|_H^2 \\ &= \|w^k-\tilde{w}^k\|_H^2-\|(w^k-\tilde{w}^k)-\alpha M(w^k-\tilde{w}^k)\|_H^2 \\ &= 2\alpha(w^k-\tilde{w}^k)^\top HM(w^k-\tilde{w}^k)-\alpha^2(w^k-\tilde{w}^k)^\top M^\top HM(w^k-\tilde{w}^k) \\ &= \alpha(w^k-\tilde{w}^k)^\top(Q^\top+Q-\alpha M^\top HM)(w^k-\tilde{w}^k) \\ &= \alpha\|w^k-\tilde{w}^k\|_G^2.\end{aligned} \tag{3.30}$$

Substituting (3.29) and (3.30) into (3.26), the assertion is proved.

With the assertion (3.24) and positive definiteness of the matrix G, we then show that the sequence $\{w^k\}$ generated by the scheme (3.1) is strictly contractive with respect to Ω^* in the following theorem.

Theorem 3.3 *Let $\{w^k\}$ be the sequence generated by the scheme (3.1); $\tilde{w}^k$, H and G be defined in (3.3) and (3.6). Then we have*

$$\|w^{k+1}-w^*\|_H^2 \le \|w^k-w^*\|_H^2-\alpha\|w^k-\tilde{w}^k\|_G^2. \tag{3.31}$$

Proof: Setting $w=w^*$ in (3.24), we obtain

$$\|w^k-w^*\|_H^2-\|w^{k+1}-w^*\|_H^2 \ge \alpha\|w^k-\tilde{w}^k\|_G^2+2\alpha\{\theta(\tilde{u}^k)-\theta(u^*)+(\tilde{w}^k-w^*)^\top F(\tilde{w}^k)\}. \tag{3.32}$$

Invoking the optimality of w^* and the monotonicity of $F(w)$, we get

$$\theta(\tilde{u}^k)-\theta(u^*)+(\tilde{w}^k-w^*)^\top F(\tilde{w}^k) \ge \theta(\tilde{u}^k)-\theta(u^*)+(\tilde{w}^k-w^*)^\top F(w^*) \ge 0, \tag{3.33}$$

and thus

$$\|w^k-w^*\|_H^2-\|w^{k+1}-w^*\|_H^2 \ge \alpha\|w^k-\tilde{w}^k\|_G^2. \tag{3.34}$$

The assertion (3.31) is arrived.

Finally, the global convergence of the scheme (1.6) can be obtained in the following theorem.

Theorem 3.4 *The sequence $\{w^k\}$ is generated by the proposed method converges to a solution of (1.1).*

Proof: First, according to (3.31), it holds that the sequence $\{w^k\}$ is bounded and

$$\lim_{k\to\infty}\|w^k-\tilde{w}^k\|_G^2 \to 0. \tag{3.35}$$

Therefore $\{w^k\}$ (and $\{\tilde{w}^k\}$) has a cluster point w^∞ and a subsequence $\{w^{k_j}\}$ such that $w^{k_j}\to w^\infty$ as $j\to\infty$. Using (3.31) again, we get $w^k\to w^\infty$ (and $\tilde{w}^k\to w^\infty$) as $k\to\infty$. Then, it follows from (3.5) and (3.35) that

$$\theta(u)-\theta(u^\infty)+(w-w^\infty)^\top F(w^\infty)\ge 0, \quad \forall w\in\Omega, \tag{3.36}$$

and thus w^∞ is a solution point of $VI(\Omega,F,f)$. The proof is complete.

3.3 Convergence rate in the ergodic sense

In this subsection, we establish the worst-case convergence rate of proposed algorithm measured by the iteration complexity in the ergodic sense for the scheme (3.1).

For the convergence rate analysis, we recall the characterization of the solution set of $VI(\theta,F,f)$, and describe it in the following theorem whose proof can be found in [FacchineiPang2003] (Theorem 2.3.5).

Theorem 3.5 *The solution set of $VI(\theta,F,f)$ is convex and it can be characterized as*

$$\Omega^*=\bigcap_{w\in\Omega}\{\tilde{w}\in\Omega\ :\ \theta(u)-\theta(\tilde{u})+(w-\tilde{w})^\top F(w)\ge 0\}. \tag{3.37}$$

With given $\epsilon > 0$, $\tilde{w}$ is called an ϵ-approximate solution of $VI(\theta, F, f)$ if it satisfies

$$\theta(u) - \theta(\tilde{u}) + (w - \tilde{w})^\top F(w) \geq -\epsilon, \quad \forall \epsilon \in D(\tilde{w}), \tag{3.38}$$

where $D(\tilde{w}) = \{w \in \Omega \mid \|w - \tilde{w} \leq 1\|\}$.

Therefore, we only need to show that for given $\epsilon > 0$, based on t iterations of the scheme (3.1), we can find $\tilde{w} \in W$ such that

$$\sup_{w \in D(\tilde{w})} \{\theta(\tilde{u}) - \theta(u) + (\tilde{w} - w)^\top F(w)\} \leq \epsilon, \tag{3.39}$$

This indicates the worst-case $O(1/t)$ convergence rate is established for the scheme (3.1).

We establish the worst-case $O(1/t)$ convergence rate for the scheme (3.1) in the following theorem.

Theorem 3.6 *Let $\{w^k\}$ be the sequence generated by the scheme (3.1). Let $\{\tilde{w}^k\}$ be defined in (3.3), and H be defined in (3.14), respectively. For any integer $t > 0$, let*

$$\tilde{w}_t = \frac{1}{t+1} \sum_{k=0}^{t} \tilde{w}^k. \tag{3.40}$$

Then we have $\tilde{w}_t \in \Omega$ and

$$\theta(\tilde{u}_t) - \theta(u) + (\tilde{w}_t - w)^\top F(w) \leq \frac{1}{2t} \|w - w^0\|_H^2, \quad \forall w \in \Omega. \tag{3.41}$$

Proof: Invoking the monotonicity of $F(w)$, we have

$$(w - \tilde{w}^k)^\top F(w) \geq (w - \tilde{w}^k)^\top F(\tilde{w}^k). \tag{3.42}$$

Substituting it into (3.24), and noting $G \succ 0$, we obtain

$$\theta(u) - \theta(\tilde{u}^k) + (w - \tilde{w}^k)^\top F(w) \geq \frac{1}{2\alpha}(\|w - w^{k+1}\|_H^2 - \|w - w^k\|_H^2), \quad \forall w \in \Omega. \tag{3.43}$$

Summarizing the above inequality over $k = 0, 1, ..., t$, and using the convexity of Ω, we obtain $\tilde{u}_t \in \Omega$ and

$$(t+1)\theta(u) - \sum_{k=0}^{t} \theta(\tilde{u}^k) + ((t+1)w - \sum_{k=0}^{t} \tilde{w}^k)^\top F(w) \geq -\frac{1}{2\alpha} \|w - w^0\|_H^2, \quad \forall w \in \Omega. \tag{3.44}$$

With the notation $\tilde{w}_t$, the above inequality can be written as

$$\frac{1}{t+1} \sum_{k=0}^{t} \theta(\tilde{u}^k) - \theta(u) + (\tilde{w}_t - w)^\top F(w) \leq \frac{1}{2(t+1)\alpha} \|w - w^0\|_H^2, \quad \forall w \in \Omega. \tag{3.45}$$

Note $\theta(u)$ is convex and $\tilde{u}_t = \frac{1}{t+1} \sum_{k=0}^{t} \tilde{u}^k)$, we have

$$\theta(\tilde{u}_t) \leq \frac{1}{t+1} \sum_{k=0}^{t} \theta(\tilde{u}^k). \tag{3.46}$$

Substituting the above inequality into (3.45), the assertion is proved.

The above theorem indicates that the average of the first t iterates generated by the scheme (3.1) is an approximate solution of $VI(\theta, F, f)$ with an accuracy of $O(1/t)$. This means the worst-case $O(1/t)$ convergence rate measured by the iteration complexity in the ergodic sense.

4 Numerical Experiments

In this section, we investigate the performance of the proposed algorithm for solving a class of sparse matrix minimization problems. All the algorithms are coded and simulated in MATLAB R2015a on a PC with Intel Core i7 CPU at 3.6GHz with 8 GB memory.

4.1 Numerical results on synthetic problem

In this subsection, we investigate the performance of our proposed algorithm for solving the following linear constrained quadratic programming (LCQP):

$$\begin{aligned} \min_{x} \quad & f_1(x_1) + ... + f_m(x_m) \\ s.t. \quad & A_1x_1 + ... + A_mx_m = c, \end{aligned} \tag{4.1}$$

where $f_i(x_i) = \frac{1}{2}x_i^\top H_i x_i + x_i^\top q_i (i = 1, ..., m) \in \mathcal{R}^{m_i} \to \mathcal{R}$, $A_i \in \mathcal{R}^{n\times m_i}$.

For ease of notation, we denote

$$f(x) = \sum_{i=1}^{m} f_i(x_i), x = \begin{pmatrix} x_1 \\ \vdots \\ x_m \end{pmatrix}, w = \begin{pmatrix} x \\ \lambda \end{pmatrix}, A = \begin{pmatrix} A_1 & \cdots & A_m \end{pmatrix}. \tag{4.2}$$

To show the efficiency of the proposed algorithm, we carry out the numerical comparison of solving the problem (4.1) by comparing the new algorithm with two other methods: the Block-wise ADMM with Relaxation factor [20] (denoted by BADMMR); the Generalized Symmetric ADMM [2] (denoted by GSADMM).

As all the test algorithms are basically block-wise ADMM, in order to implement the algorithms, we need to regroup the variables of (4.1). For our algorithm, when $m = 3$, we can group the variables by two ways: "1∼2" or "2∼1"; when $m \geq 4$, the grouping strategies can be "(m-3)∼3", "(m-2)∼2" or "(m-1)∼1". For the other two test algorithms, there is no constraint on the grouping strategy, i.e., the grouping strategy can be "1∼(m-1)",...,"(m-1)∼1".

To restrict the capacity, we only consider and test the case with $m = 4$. In this case, by three different ways of grouping the variables of (4.1), the proposed algorithm can lead to the following three algorithms:

$$\begin{cases} \bar{x}_1^k = \arg\min\{\frac{1}{2}x_1^\top H_1x_1 + x_1^\top q_1 + \langle\Lambda^k, A_1x_1\rangle + \frac{\beta}{2}\|A_1x_1 + A_2x_2^k + A_3x_3^k + A_4x_4^k - c\|_F^2 + \frac{\tau\beta}{2}\|A_1(x_1 - x_1^k)\|_F^2\}, \\ \bar{x}_2^k = \arg\min\{\frac{1}{2}x_2^\top H_2x_2 + x_2^\top q_2 + \langle\Lambda^k, A_2x_2\rangle + \frac{\beta}{2}\|A_1x_1^k + A_2x_2 + A_3x_3^k + A_4x_4^k - c\|_F^2 + \frac{\tau\beta}{2}\|A_2(x_2 - x_2^k)\|_F^2\}, \\ \bar{x}_3^k = \arg\min\{\frac{1}{2}x_3^\top H_3x_3 + x_3^\top q_3 + \langle\Lambda^k, A_3x_3\rangle + \frac{\beta}{2}\|A_1x_1^k + A_2x_2^k + A_3x_3 + A_4x_4^k - c\|_F^2 + \frac{\tau\beta}{2}\|A_3(x_3 - x_3^k)\|_F^2\}, \\ \bar{x}_4^k = \arg\min\{\frac{1}{2}x_4^\top H_4x_4 + x_4^\top q_4 + \langle\Lambda^k, A_4x_4\rangle + \frac{\beta}{2}\|A_1\bar{x}_1^k + A_2\bar{x}_2^k + A_3\bar{x}_3^k + A_4x_4 - c\|_F^2, \\ \bar{\lambda}^k = \lambda^k - \beta(A\bar{x}^k - c), \\ w^{k+1} = w^k - \alpha(w^k - \bar{w}^k), \quad \tau > 2, \quad \alpha \in (0, 1). \end{cases} \tag{4.3}$$

$$\begin{cases} \bar{x}_1^k = \arg\min\{\frac{1}{2}x_1^\top H_1x_1 + x_1^\top q_1 + \langle\Lambda^k, A_1x_1\rangle + \frac{\beta}{2}\|A_1x_1 + A_2x_2^k + A_3x_3^k + A_4x_4^k - c\|_F^2 + \frac{\tau\beta}{2}\|A_1(x_1 - x_1^k)\|_F^2\}, \\ \bar{x}_2^k = \arg\min\{\frac{1}{2}x_2^\top H_2x_2 + x_2^\top q_2 + \langle\Lambda^k, A_2x_2\rangle + \frac{\beta}{2}\|A_1x_1^k + A_2x_2 + A_3x_3^k + A_4x_4^k - c\|_F^2 + \frac{\tau\beta}{2}\|A_2(x_2 - x_2^k)\|_F^2\}, \\ \bar{x}_3^k = \arg\min\{\frac{1}{2}x_3^\top H_3x_3 + x_3^\top q_3 + \langle\Lambda^k, A_3x_3\rangle + \frac{\beta}{2}\|A_1\bar{x}_1^k + A_2\bar{x}_2^k + A_3x_3 + A_4x_4^k - c\|_F^2\}, \\ \bar{x}_4^k = \arg\min\{\frac{1}{2}x_4^\top H_4x_4 + x_4^\top q_4 + \langle\Lambda^k, A_4x_4\rangle + \frac{\beta}{2}\|A_1\bar{x}_1^k + A_2\bar{x}_2^k + A_3x_3^k + A_4x_4 - c\|_F^2, \\ \bar{\lambda}^k = \lambda^k - \beta(A\bar{x}^k - c), \\ w^{k+1} = w^k - \alpha(w^k - \bar{w}^k), \quad \tau > 1, \quad \alpha \in (0, 2 - \sqrt{2}). \end{cases} \tag{4.4}$$

$$\begin{cases} \bar{x}_1^k = \arg\min\{\frac{1}{2}x_1^\top H_1x_1 + x_1^\top q_1 + \langle\Lambda^k, A_1x_1\rangle + \frac{\beta}{2}\|A_1x_1 + A_2x_2^k + A_3x_3^k + A_4x_4^k - c\|_F^2 + \frac{\tau\beta}{2}\|A_1(x_1 - x_1^k)\|_F^2\}, \\ \bar{x}_2^k = \arg\min\{\frac{1}{2}x_2^\top H_2x_2 + x_2^\top q_2 + \langle\Lambda^k, A_2x_2\rangle + \frac{\beta}{2}\|A_1\bar{x}_1^k + A_2x_2 + A_3x_3^k + A_4x_4^k - c\|_F^2\}, \\ \bar{x}_3^k = \arg\min\{\frac{1}{2}x_3^\top H_3x_3 + x_3^\top q_3 + \langle\Lambda^k, A_3x_3\rangle + \frac{\beta}{2}\|A_1\bar{x}_1^k + A_2x_2^k + A_3x_3 + A_4x_4^k - c\|_F^2\}, \\ \bar{x}_4^k = \arg\min\{\frac{1}{2}x_4^\top H_4x_4 + x_4^\top q_4 + \langle\Lambda^k, A_4x_4\rangle + \frac{\beta}{2}\|A_1\bar{x}_1^k + A_2x_2^k + A_3x_3^k + A_4x_4 - c\|_F^2, \\ \bar{\lambda}^k = \lambda^k - \beta(A\bar{x}^k - c), \\ w^{k+1} = w^k - \alpha(w^k - \bar{w}^k), \quad \tau > 0, \quad \alpha \in (0, 2 - \sqrt{3}). \end{cases} \tag{4.5}$$

Table 1: Numerical results on synthetic problem.

(n, m_i)	Grouping	BADMMR			GSADMM			New Algorithm		
		iter	time	KKT	iter	time	KKT	iter	time	KKT
(100,50)	1∼3	1871.0	0.163	3.938e-8	1537.4	0.135	2.420e-8	2000.0	0.176	7.626e-2
	2∼2	1887.3	0.163	7.253e-8	1537.3	0.134	2.032e-8	934.7	0.082	1.177e-8
	3∼1	207.7	0.018	2.627e-9	161.0	0.014	2.105e-9	333.0	0.026	4.397e-9
(100,100)	1∼3	498.6	0.095	1.041e-8	402.7	0.076	8.188e-9	2000.0	0.381	1.185e-5
	2∼2	477.7	0.090	1.016e-8	403.7	0.076	8.032e-9	254.3	0.048	4.895e-9
	3∼1	230.7	0.045	3.448e-9	231.8	0.044	3.334e-9	220.7	0.040	3.368e-9
(50,100)	1∼3	194.5	0.033	6.294e-9	153.9	0.025	4.760e-9	1216.7	0.193	2.114e-4
	2∼2	181.0	0.027	6.895e-9	146.1	0.022	5.263e-9	129.2	0.020	3.031e-9
	3∼1	435.7	0.067	4.985e-9	435.4	0.068	4.845e-9	410.5	0.060	5.067e-9

Note that all the subproblems in (4.3), (4.4) and (4.5) are unconstrained quadratic programming (QP) which can be efficiently solved by PCG. Furthermore, since the Hessian in each subproblem is fixed, we can do a cholesky decomposition on the Hessian at first, then the computation of subproblems at each iteration can be decomposed into two simpler QP (as the Hessian are either upper triangle or lower triangle).

The test problem is randomly generated in which all entries in H, q, A and c are i.i.d. Gaussian. For all the test algorithms, the maximal number of iterations is set to be 2000, the stopping tolerance is set to be 10^{-10}, the initial values of working variables are set as zero vectors, and the stopping criterion is as follows:

$$\text{relchg}(k) = \max\{\|x_1^k - x_1^{k-1}\|/\|x_1^{k-1}\|, ..., \|x_m^k - x_m^{k-1}\|/\|x_m^{k-1}\|, \|\lambda^k - \lambda^{k-1}\|/\|\lambda^{k-1}\|\} \leq \text{tol}. \tag{4.6}$$

The setting of key parameter β is critical to the performance of test algorithms, and its optimal setting can be different with different problem settings, hence its setting is tuned by hand. For our algorithm, the parameter τ and the extension factor α are always chosen to be close to its upper bound (or lower bound), e.g., in (4.4), we set $\tau = 1.01, \alpha = 0.58$. For BADMMR which needs to apply an extension factor to the λ update, we set it to be close to its upper bound $((\sqrt{5}+1)/2)$, e.g., 1.6. For GSADMM which requires two extension factors on two updates of multipliers, we set them to be 0.9 and 1.09 as suggested in [Bai 2018a].

We compare the test algorithms from three aspects: number of iterations, computation time and accuracy. As we do not know the true solution of the underlying problem, instead, we use the KKT violation as a surrogate of the accuracy. The KKT violation at k-th iteration is defined as follows:

$$KKT(k) := \max(\|KKT_1(k)\|, ..., \|KKT_m(k)\|, \|KKT_\lambda(k)\|), \tag{4.7}$$

where

$$KKT_i(k) = H_i x_i^k + q_i - A_i^\top \lambda^k, \quad i = 1, ..., m, \quad KKT_\lambda(k) = A_1 x_1^k + ... + A_m x_m^k - c. \tag{4.8}$$

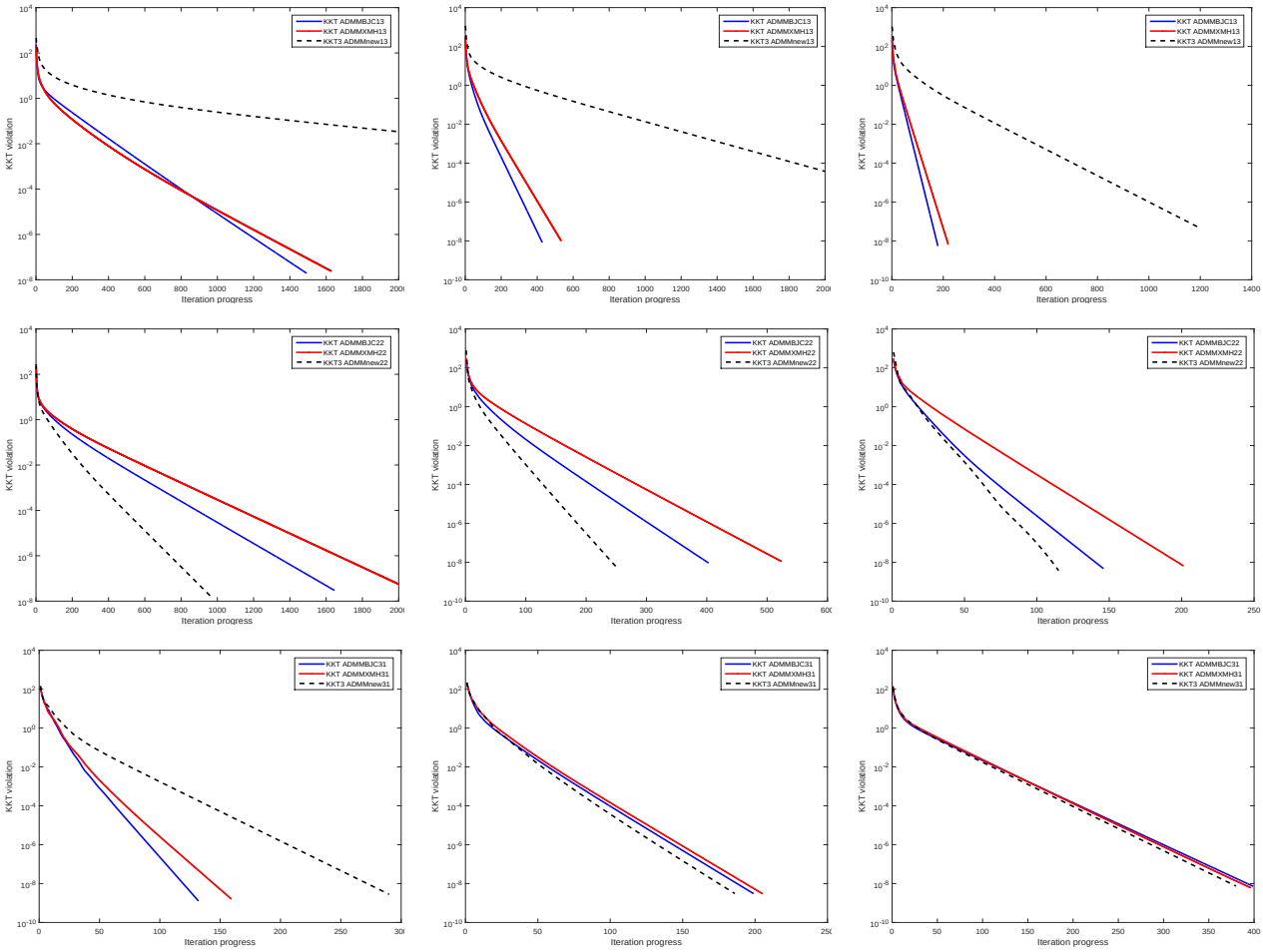
The results obtained by the test algorithms under several different problem settings are reported in Table 1, in which "iter" and "time" denote the number of iteration and computation time (in seconds), respectively.

We observe from Table 1 that the test algorithms can behave differently with different problem settings, so we can not determine which one is always the best. Then we find that the new algorithm outperforms the other two algorithms in 5 cases among 9 cases. This may indicate that our algorithm is overall more efficient than the other two algorithms, and its performance advantage seems to be more prominent with smaller setting of n/m_i. We also find that the grouping strategy affect the performance much. For instance, when (n, m_i)=(100,50) or (100,100), the 3∼1 strategy seems to be better, while the 2∼2 strategy is more preferable when (n, m_i)=(50,100). The 1∼3 strategy is usually the worst one, and this is especially true for our algorithm.

To investigate the convergence behavior of the test algorithms in a more precise way, The iteration progress of KKT violations with several different problem settings are plotted in Figure 1.

The results shown in Figure 1 coincide with that shown in Table 1. For instance, when the grouping strategy is 1∼3, our algorithm converge very slow; when the grouping strategy is 2∼2, our algorithm always

Figure 1: From left to right are the results with (n, m_i)=(100,50), (100,100) and (50,100), respectively. From above to below are the results with 1∼3, 2∼2, and 3∼1 grouping strategy, respectively.



outperforms the other two algorithms, and the performance gap is obvious. As for the 3∼1 strategy, the numerical results are a little tricky: our algorithm performs slightly better than the other two algorithms under two problem settings, while the other two algorithms perform significantly better than our algorithm under the other one problem setting ((n, m_i)=(100,50)).

4.2 Numerical results on a practical problem

In this subsection, we consider the robust principal component analysis (RPCA) problem which aims to recover one low-rank and one sparse matrices from their sum. This problem arises from various areas such as the model selection in statistics, image processing, etc [5, 28]. Specially, the following convex surrogate model of the RPCA problem is usually adopted:

$$\begin{aligned} \min_{A,E} \quad & \|A\|_* + \mu\|E\|_1 \\ s.t. \quad & A + E = C, \end{aligned} \tag{4.9}$$

where $C \in \mathcal{R}^{m\times n}$ is the data matrix. The nuclear norm $\|\cdot\|_*$ is a convex surrogate which can catch the low-rank component of C while the elementwise l1 norm $\|\cdot\|_1$ is to induce the sparse component of C. It has been verified that, under certain mild assumptions, the model (4.9) can recover the original solution accurately.

When the observation is corrupted by Gaussian noise, the solution of (4.9) may not be accurate, then the

following model was suggested [26]:

$$\begin{aligned} \min_{A,E} \quad & \|A\|_* + \mu\|E\|_1 \\ s.t. \quad & A + E + Z = C, \quad \|Z\|_F \leq \delta, \end{aligned} \tag{4.10}$$

where the parameter $\delta > 0$ depends on the noise level and $\|\cdot\|_F$ is the Frobenius norm.

We generate A by $A = UV^\top$, where $U \in \mathcal{R}^{m\times k}$ and $V \in \mathcal{R}^{k\times n}$ are randomly generated whose entries are i.i.d. Gaussian. The sparse matrix E is randomly generated whose non-zero entries are i.i.d. uniformly in the interval [-50, 50]. Let SR denotes the ratio of non-zero entries (i.e., $\|E\|_0/mn$). The noise matrix Z is randomly generated with i.i.d. Gaussian, then it is scaled such that $\|Z\|_F = \eta\|A + E\|_F$ where η is the noise level parameter.

It is easy to observe that model (4.10) is a special case of the general model (1.1) with 3 blocks of variables, hence we can use our proposed method to solve it in either $1 \sim 2$ or $2 \sim 1$ fashion (denoted by ADMM12 and ADMM21, respectively). To reveal the efficiency of the proposed method, some efficient algorithms are included in our comparison: the Generalized Symmetric ADMM [2] in $1 \sim 2$ or $2 \sim 1$ fashion (denoted by GSADMM12 and GSADMM21, respectively); the splitting method [19] (denoted by ADMMHTY) which is a special case of Block-wise ADMM [4]; the partial splitting augmented Lagrangian method (denoted by PSAL). The subproblems involved in these algorithms (including our algorithm) can be solved explicitly: the A-subproblem can be solved by a partial singular value decomposition (SVD); the E-subproblem can be solved by the soft shrinkage operation; the Z-subproblem can be computed by a simple projection. The main cost ar each iteration lies in the partial SVD. Note there have been a bunch of efficient partial SVD solvers available (e.g., LMSVD, SLRP, ...), however, as reducing the per-iteration cost is not our main concern, we still executed the partial SVD by implementing the package of PROPACK for all test algorithms. Some necessary parameters are set as follows: $\mu = 3$, $\eta = 0.01$, $\delta = 0.001$, $\beta = 0.35$; τ and α are set to be close to their boundaries, e.g., we set $\tau = 1.01$, $\alpha = 0.99$ in our algorithm with $2 \sim 1$ fashion.

We compare the test algorithms from the aspects of computational cost and solution accuracy. To measure the solution accuracy, we use the relative error of A and E with their accurate solution defined as follows:

$$err(A) := \frac{\|A^k - A^*\|_F}{\|A^*\|_F} \quad \text{and} \quad err(E) := \frac{\|E^k - E^*\|_F}{\|E^*\|_F}. \tag{4.11}$$

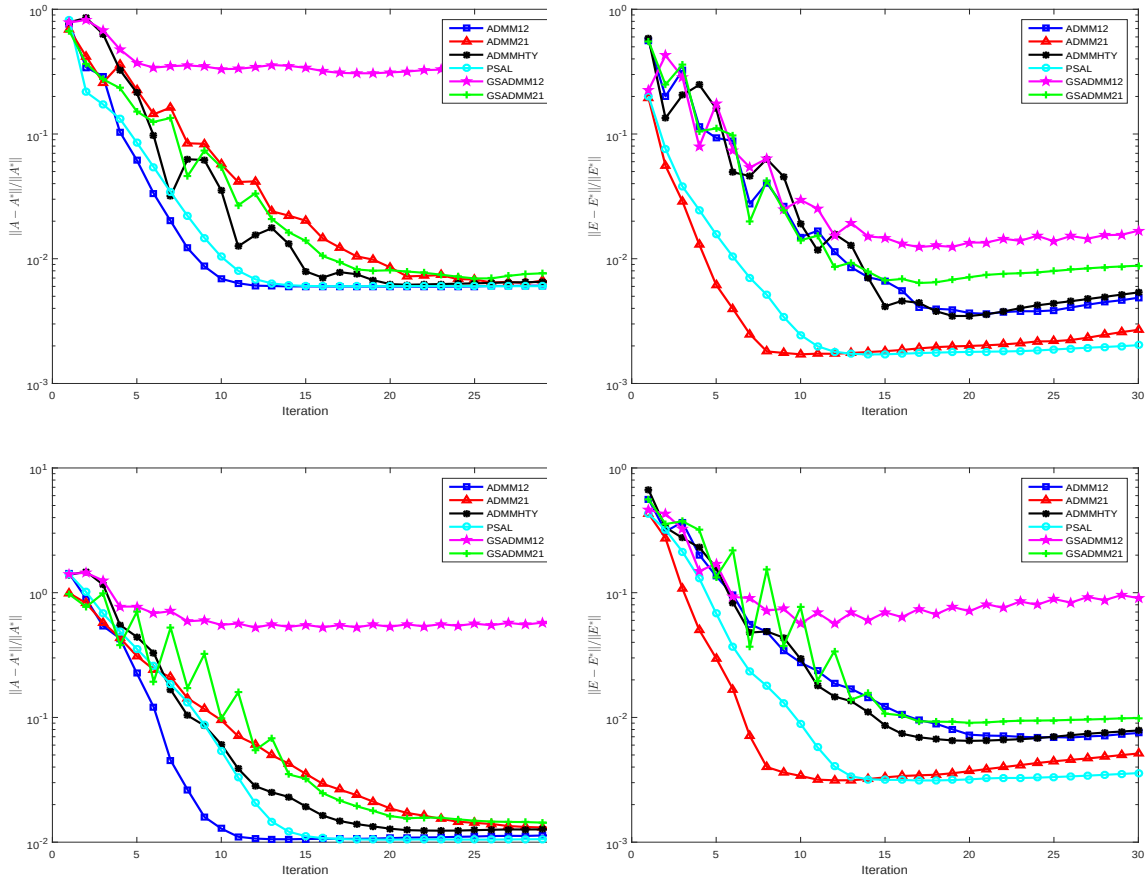
We carried out the experiments with several problem settings. All variables in the test algorithms were initialized with random matrices whose entries are i.i.d. Gaussian. We let all algorithms run for fixed 30 iterations, and the iteration progress of relative error of the recovered low-rank and sparse matrices are reported in Figure 2 and 3. We observed that all the test algorithms can achieve similar accuracy for each tested scenario (only except GSADMM12), while their convergence speed can be different. In terms of $err(A)$, ADMM12 is the fastest one among all test algorithms; while in terms of $err(E)$, ADMM21 is faster than other algorithms. This observation can guide us to select the proper grouping fashion: when the underlying problem is image denoising or video separation where the low-rank matrix A is our major concern, we should adopt the $1\sim2$ fashion; when the problem under consideration is recover a sparse data matrix, the $2\sim1$ fashion is more preferable.

We can draw some preliminary conclusions as follows: (1) our algorithm is competitive compared with some most efficient existing algorithms; (2) the performance of our algorithm can be very different under different grouping strategy, and this can guide us to select the best fashion for the specific problem; (3) for synthetic problem, our algorithm performs better with smaller setting of n/m_i which is true in many practical applications, e.g., compressive sensing, hence our algorithm should be more suitable for solving this practical problems.

5 Conclusions

There has been a constantly increasing interest in developing and improving the theory of the ADMM for solving the multi-block separable convex optimization in recent years. We verify its numerical advantages by both synthetic and practical problems. In this paper, we propose a new partial PPA block-wise ADMM, and the contribution is two-fold. From the theoretical aspect, compared with existing multi-block ADMM,

Figure 2: Iteration progress of relative error $(m, n) = (100, 100)$. First row: $k = 2$, $SR = 2\%$; Second row: $k = 6$, $SR = 6\%$; .



we require less subproblems to be regularized by proximal terms which might imply better quality of the resulting sequence. From the practical aspect, our algorithm is competitive compared with some state-of-the-art algorithms for both synthetic and practical problems, and its performance can be grouping fashion dependant, i.e., its numerical advantage can be especially prominent when the grouping fashion is properly chosen.

References

[1] J.C. Bai and H.C. Zhang. A one-parameter family of middle proximal admm for constrained separable convex optimization. *submitted*, 2018.

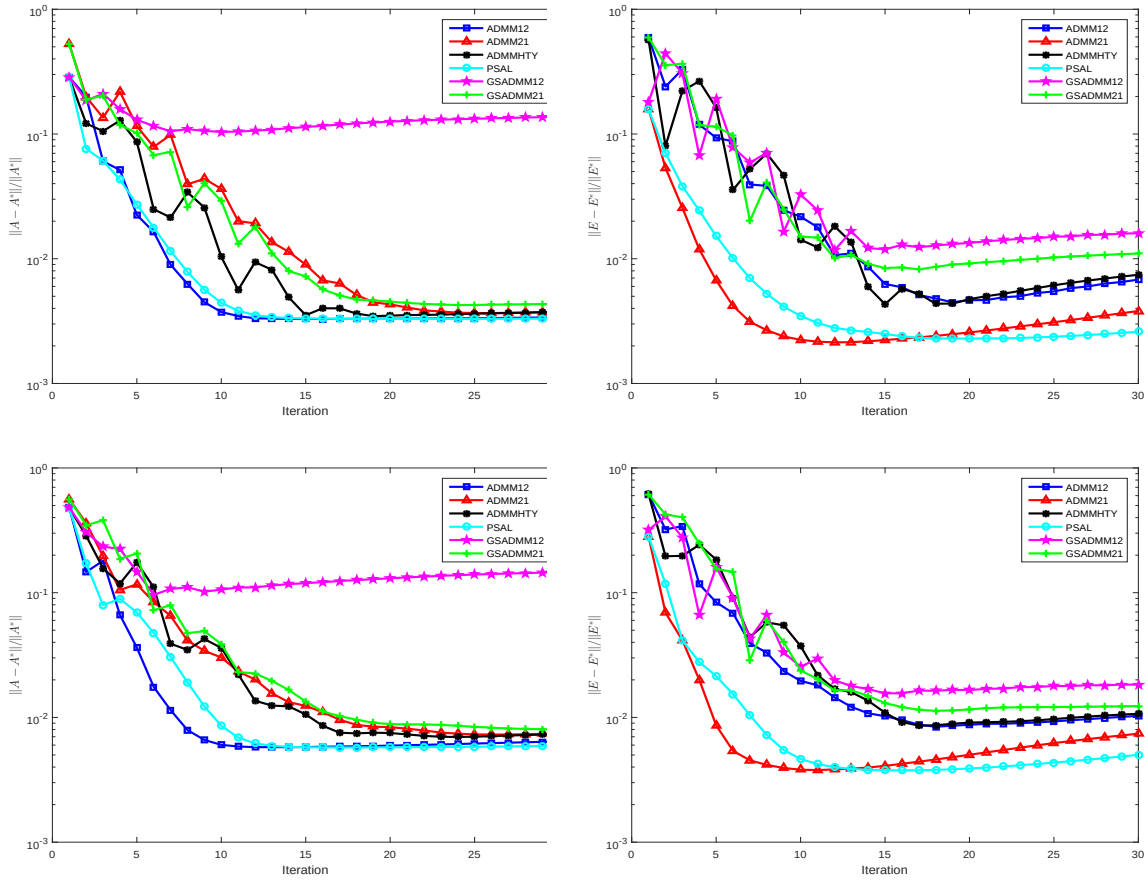
[2] J.C. Xu F.M. Bai, J.C. Li and H.C. Zhang. Generalized symmetric admm for separable convex optimization. *Comput. Optim. Appl.*, 70:129–170, 2012.

[3] J. Bien and R.J. Tibshirani. Sparse estimation of a covariance matrix. *Biometrika*, 98(4):807–820, 2011.

[4] He B.S. and X.M. Yuan. Block-wise alternating direction method of multipliers for multiple-block convex programming and beyond. *SMAI J. Comput. Math.*, 1:145–174, 2015.

[5] E. J. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *J. ACM*, 58(3):Article No. 11, 2009.

Figure 3: Iteration progress of relative error $(m, n) = (500, 500)$. First row: $k = 10$, $SR = 2\%$; Second row: $k = 30$, $SR = 6\%$; .



[6] P.J. Chang X.K. Liu S.Y. Zhao and X. Li. Convergent prediction-correction-based admm for multi-block separable convex programming. *J. Comput. Appl. Math.*, 335:270–288, 2018.

[7] B.S. Ye Y.Y. Chen, C.H. He and X.M. Yuan. The direct extension of admm for multi-block convex minimization problems is not necessarily convergent. *Math. Program.*, 155(1):57–79, 2016.

[8] C. Chen, Y. Shen, and Y. You. On the convergence analysis of the alternating direction method of multipliers with three blocks. *Abstr. Appl. Anal.*, 2013, Article ID 183961:7 pages, 2013.

[9] R. Glowinski and A. Marrocco. Sur lapproximation par elements finis dordre un, et la resolution par penalisation-dualite dune classe de problemes de dirichlet nonlineaires. *Rev. Francaise dAut. Inf. Rech. Oper., R-2*, pages 41–76, 1975.

[10] D.R. Han and X.M. Yuan. A note on the alternating direction method of multipliers. *J. Optim. Theory Appl.*, 155(1):227–238, 2012.

[11] H.J. Yang H. Han, D.R. He and X.M. Yuan. A customized douglas-rachford splitting algorithm for separable convex minimization with linear constraints. *Numer. Math.*, 127(1):167–200, 2014.

[12] W.W. Han, D.R. Kong and W.X. Zhang. A partial splitting augmented lagrangian method for low patch-rank image decomposition. *J. Math. Imaging Vis.*, 51:145–160, 2015.

[13] X.M. Han, D.R. Yuan and W.X. Zhang. An augmented lagrangian based parallel splitting method for separable convex minimization with applications to image processing. *Math. Comput.*, 83:2263–2291, 2014.

[14] B.S. He. Parallel splitting augmented lagrangian methods for monotone structured variational inequalities. *Comput. Optim. Appl.*, 42:195–212, 2009.

[15] B.S. He and X.M. Yuan. On the o(1/n) convergence rate of the douglas-rachford alternating direction method. *SIAM J. Numer. Anal.*, 50(2):700–709, 2012.

[16] L.S. He, B.S. Hou and X.M. Yuan. On full jacobian decomposition of the augmented lagrangian method for separable convex programming. *SIAM J. Opt.*, 25(4):2274–2312, 2015.

[17] M. Xu M.H. He, B.S. Tao and X.M. Yuan. An alternating direction-based contraction method for linearly constrained separable convex programming problems. *Optimization*, 62(4):573–596, 2013.

[18] M.and Yuan X.M. He, B.S. Tao. Alternating direction method with gaussian back substitution for separable convex programming. *SIAM J. Optim.*, 22(2):313–340, 2012.

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

[20] M.H. He, B.S. Xu and Yuan X.M. Block-wise admm with a relaxation factor for multiple-block convex programming. *J. Oper. Res. Soc. China*, accepted, 2018.

[21] M. R. Hestenes. Multiplier and gradient methods. *J. Optim. Theory Appl.*, 4:303–320, 1969.

[22] M. Hong and Z. Luo. On the linear convergence of the alternating direction method of multipliers. *Arxiv preprint,https://arxiv.org/abs/1208.3922*, 2013.

[23] H.J. Hou, L.S. He and J.F. Yang. A partially parallel splitting method for multiple-block separable convex programming with applications to robust pca. *Comput. Optim. Appl.*, 63(1):273–303, 2016.

[24] J.C. Li G. Bai J.C.and Liu X.N. Liu, Z.S. Li. A new model for sparse and low-rank matrix decomposition. *J. Appl. Anal. Comput.*, 7(2):600–616, 2017.

[25] Z.W. Shen, Y. Wen and Y. Zhang. Augmented lagrangian alternating direction method for matrix separation based on low-rank factorization. *Optim. Methods & Softw.*, 29(2):239–263, 2014.

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

[27] J.J. Wang and Song W. An algorithm twisted from generalized admm for multi-block separable convex minimization models. *J. Comput. Appl. Math.*, 309(c):342–358, 2017.

[28] J. Wright, A. Ganesh, S. Rao, Y. Peng, and Y. Ma. Robust principal component analysis: Exact recovery of corrupted low-rank matrices by convex optimization. In *In Proceedings of Neural Information Processing Systems (NIPS)*, December 2009.

[29] Y.Y. Xu. Hybrid jacobian and gauss–seidel proximal block coordinate update methods for linearly constrained convex programming. *SIAM J. Opt.*, 28(1):646–670, 2018.

[30] J. Yang and Y. Zhang. Alternating direction algorithms for ℓ_1-problems in compressive sensing. *SIAM J. Sci. Comput.*, 33(1):250–278, 2011.