A New Sequential Updating Scheme of the Lagrange Multiplier for Multi-Block Linearly Constrained Separable Convex Optimization with Relaxed Step Sizes

Yuan Shen[†] [1] **Yannian Zuo**[†] [2] **Xiayang Zhang**[‡][3]

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

Department of Mathematics and Physics, Nanjing Institute of Technology, Nanjing 211167, P.R.China.[‡]

Abstract: In various applications such as signal/image processing, data mining, statistical learning and etc., the multi-block linearly constrained separable convex optimization is frequently used, where the objective function is the sum of multiple individual convex functions, and the major constraints are linear. A classical method for solving such kind of optimization problem could be the alternating direction method of multipliers(ADMM). It decomposes the subproblem into a series of small-scale ones such that its per-iteration cost can be very low. However, ADMM is originally designed for two-block model, and its convergence can not be guaranteed for general multi-block model without additional assumptions. Dai et al. proposed a variant of ADMM entitled sequential updating scheme of the Lagrange multiplier (SUSLM), in which the multiplier is updated multiple times at each iteration. In order to derive its convergence property, a correction step is imposed at the end of each iteration. In this paper, we improve the SUSLM by introducing two parameters hence the condition of step sizes is relaxed. Preliminary experimental results show that the new algorithm converges faster than the original SUSLM algorithm, and it is empirically effective on solving both synthetic and real problems compared with several efficient ADMM-based algorithms.

Keywords: alternating direction method of multipliers, augmented Lagrangian, multi-block, sequential updating scheme of the lagrange multiplier.

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

1 Introduction

In this paper, we consider the following multi-block convex optimization with separable objective function and linear constraints:

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

where $f_i(x_i) : \mathcal{R}^{m_i} \to \mathcal{R}$ are closed convex functions (may not be smooth); $\sum_{i=1}^{p} m_i = m$; $A_i \in \mathcal{R}^{n \times m_i}$ and $c \in \mathcal{R}^n$; $\mathcal{X}_i \subseteq \mathcal{R}^{m_i}$ are closed convex sets; $p \geq 3$. Throughout this paper, the solution set of the problem (1.1) is assumed to be nonempty. In addition, it is mildly assumed that all the matrices $A_i, i = 1, ..., p$ have full column rank. For simplicity, we introduce the notations as follows: $A = (A_1, ..., A_p)$, $x = (x_1^\top, ..., x_p^\top)^\top$, $\mathcal{X} = \mathcal{X}_1 \times ... \times \mathcal{X}_p$ and $\Omega = \mathcal{X} \times \mathcal{R}^n$.

[1] Email: ocsiban@126.com. Research supported by National Natural Science Foundation of China grants 11726618 and by National Social Science Foundation of China under Grant 17BTQ063 and by Social Science Foundation of Jiangsu province under Grant 18GLA002.

[2]Email: 1059665104@qq.com.

[3](Corresponding author)Email: 369318324@qq.com.

Due to its important applications in a wide variety of fields, including, but not limited to, compressive sensing problem in signal processing [32], low rank and sparse representations [27] in image processing, sparse inverse covariance estimation problem [3] in finance or statistics and so forth, the problem (1.1) has been extensively studied in the past decade.

A classical effective method for solving (1.1) is the augmented Lagrangian method (ALM) [24] whose iterative scheme is described as the following procedure:

$$\begin{cases} (x_1^{k+1}, ..., x_p^{k+1}) = \arg\min\{L_\beta(x_1, ..., x_p, \lambda^k) \mid x_i \in \mathcal{X}_i\}, \\ \lambda^{k+1} = \lambda^k - \beta((\sum_{i=1}^p A_i x_i^{k+1}) - c), \end{cases} \tag{1.2}$$

where the augmented Lagrangian (AL) function $L_\beta(X, \lambda)$ is defined as follows:

$$L_\beta(x_1, ..., x_p, \lambda) := \sum_{i=1}^p f_i(x_i) - \langle \lambda, (\sum_{i=1}^p A_i x_i^{k+1}) - c\rangle + \frac{\beta}{2}\|(\sum_{i=1}^p A_i x_i^{k+1}) - c\|^2, \tag{1.3}$$

with $\lambda \in \mathcal{R}^n$ being the Lagrange multiplier and $\beta > 0$ being a properly chosen penalty parameter.

Due to its satisfactory convergence speed, ALM is suitable for solving small-scale problems, however, when the dimension is large, minimizing the AL function could be impractical or impossible. To this end, the alternating direction method of multipliers (ADMM) [9, 8] was introduced as a splitting variation of ALM. At each iteration, ADMM sequentially minimizes the AL function over each block variable with other block variables fixed, and then an updating procedure of the multiplier is followed. For solving model (1.1), the iterative scheme of ADMM consists of following procedure:

$$\begin{cases} x_i^{k+1} = \arg\min\{L_\beta(x_1^{k+1}, ..., x_{i-1}^{k+1}, x_i, x_{i+1}^k, ..., x_p^k, \lambda^k) \mid x_i \in \mathcal{X}_i\}, \ i = 1, ..., p, \\ \lambda^{k+1} = \lambda^k - \beta((\sum_{i=1}^p A_i x_i^{k+1}) - c). \end{cases} \tag{1.4}$$

As the dimension of subproblems of ADMM can be much smaller than that of ALM, hence its per-iteration cost could be much lower. Moreover, in some specific real applications, the objective function of (1.1) may have some special properties such as separability, then the subproblems can be simple enough to have closed-form solution. ADMM always uses the newest information when it solves each subproblem, hence it is basically a Gauss-Seidel scheme. Specially, for the two-block case ($p = 2$), ADMM is equivalent to the classical Douglas-Rachford splitting method when the dual problem of (1.1) is considered [8], and it is proved to be globally convergent with $p = 2$ [15].

In many practical applications, the numerical effectiveness of ADMM for multi-block case ($p \geq 3$) has been verified, see, e.g., [30, 28]. Under additional assumptions, e.g., see [11, 21, 25, 6], the convergence of (1.4) for multi-block case has been established. However, these assumptions such as strongly convexity of the objective functions or sufficiently small β are usually too restrictive. A counter example was also given in [5] to show that ADMM may not converge for the multi-block case without additional assumptions.

Beyond the Gauss-Seidel fashion, an alternate ADMM with Jacobian fashion was also considered where all the primal variables are updated simultaneously as follows:

$$\begin{cases} x_i^{k+1} = \arg\min\{L_\beta(x_1^k, ..., x_{i-1}^k, x_i, x_{i+1}^k, ..., x_p^k, \lambda^k) \mid x_i \in \mathcal{X}_i\}, \ i = 1, ..., p, \\ \lambda^{k+1} = \lambda^k - \beta((\sum_{i=1}^p A_i x_i^{k+1}) - c). \end{cases} \tag{1.5}$$

However, the Jacobian scheme (1.5) without modification may not converge either [14].

To derive the convergence of ADMM for multi-block case, one possible remedy is to modify the iterative scheme of ADMM. For instance, He et al. [19] proposed a ADMM-based method whose iterative scheme is described as follows:

$$\begin{cases} x_1^{k+1} = \arg\min\{L_\beta(x_1, x_2^k, ..., x_p^k, \lambda^k) \mid x_1 \in \mathcal{X}_1\}, \\ x_i^{k+1} = \arg\min\{L_\beta(x_1^{k+1}, x_2^k, ..., x_{i-1}^k, x_i, x_{i+1}^k, ..., x_p^k, \lambda^k) + \frac{\tau\beta}{2}\|A_i(x_i - x_i^k)\|^2 \mid x_i \in \mathcal{X}_i\}, \ i = 2, ..., p, \\ \lambda^{k+1} = \lambda^k - \beta((\sum_{i=1}^p A_i x_i^{k+1}) - c), \end{cases} \tag{1.6}$$

where a special proximal term in the form of $\frac{\tau\beta}{2}\|A_i(x_i - x_i^k)\|^2$ with $\tau > p-2$ is added to a part of subproblems to guarantee the global convergence. As a part of subproblems can be solved simultaneously, HTY method is

actually a mixture of Jacobi and Gauss-Seidel schemes, and its numerical performance has been verified on several practical problems. Based on HTY method, Hou et al. [26] proposed a partial parallel splitting method in which the multiplier is updated right after solving the first subproblem. This method can be considered as an application of proximal point algorithm (PPA) to the original problem, hence the updated variables at each iteration can be extended by a relaxation factor close to 2. He and Yuan [22] proposed a block-wise ADMM (BADMM) in which the primal variables are grouped into two groups and the subproblems are solved in a mixture of Jacobi and Gauss-Seidel fashions. Similar to HTY method, a proximal term similar to that in (1.6) is added to each subproblem to ensure the global convergence. The numerical efficiency of BADMM was further improved by introducing a relaxation factor up to 1.618 on the multiplier update [20]. Too many proximal terms could slow down the convergence speed, Shen et al. proposed a variant of BADMM such that the proximal terms only appear in the first group of subproblems, and its performance is also justified [29].

As the update of the Lagrange multiplier is usually much cheaper to compute than the decomposed subproblems of ADMM, an idea to improve the speed performance of ADMM is to update the Lagrange multiplier multiple times. For the case with $p = 2$, the iterative scheme is as follows:

$$\begin{cases} x_1^{k+1} = \arg\min\{L_\beta(x_1, x_2^k, \lambda^k) \mid x_1 \in \mathcal{X}_1\}, \\ \lambda^{k+\frac{1}{2}} = \lambda^k - \alpha\beta(A_1 x_1^{k+1} + A_2 x_2^k - c). \\ x_2^{k+1} = \arg\min\{L_\beta(x_1^{k+1}, x_2, \lambda^{k+\frac{1}{2}}) \mid x_2 \in \mathcal{X}_2\}, \\ \lambda^{k+1} = \lambda^{k+\frac{1}{2}} - \alpha\beta(A_1 x_1^{k+1} + A_2 x_2^{k+1} - c). \end{cases} \tag{1.7}$$

The scheme (1.7) is referred as symmetric ADMM (SADMM) as the multiplier is updated two times in a symmetrical way such that the primal variables x_1 and x_2 are treated equally. SADMM can also be interpreted as applying the classical PeacemanCRachford splitting method to the dual problem of (1.1). The attached relaxation factor $\alpha \in (0, 1)$ is to guarantee its strictly contractive properties to the solution set, see (3.20) in [16].

When the iterative scheme (1.7) is directly extended to the general case with $p \geq 3$, however, the scheme is not necessarily convergent according to [23]. Based on the similar idea with BADMM, Bai et al. proposed a generalized symmetric ADMM (GSADMM) [1], in which the primal variables are also grouped into two groups and a special proximal term is also imposed to all the subproblems. The iterative scheme of GSADMM is similar to that of its ancestor BADMM. Its major difference with BADMM lies in two aspects: 1) as a Symmetric ADMM, the multipliers are updated two times at each iteration in GSADMM while there is only one updating step in the iterative scheme of BADMM; 2) its dual step size rule is relaxed which enables faster convergence. Bai and Zhang [2] proposed a more complicated variant of ADMM entitled middle proximal ADMM (MPADMM) where the proximal terms are absent in the first and the last subproblems and there is only one multipliers update at each iteration.

Another line to derive the convergence of ADMM is to correct its output by some correction step, then some ADMM-based algorithms with convergence guarantee have been proposed either in a Gauss-Seidel fashion [17, 18, 7] or in a Jacobi fashion [13, 12, 14]. For the case with $p \leq 3$, in [13], He et. al proposed an algorithm based on ADMM in which the output of ADMM with Jacobi fashion is corrected via a special correction step. Han et. al then proposed another ADMM-based method with Gauss-Seidel fashion in [**?**]. Its correction step is relatively simple, and its numerical performance is justified via the experiments in some practical problems. However, these two methods can only adapt to case with at most three blocks variables. To this end, another ADMM-based method with Gauss-Seidel fashion was then proposed [17] which works for the case with arbitrary blocks variables. In [18], He et. al proposed another ADMM-based algorithm with Gauss-Seidel fashion entitled ADMM with Gaussian back substitution (ADMMG). It uses a standard ADMM scheme to generate the predictor, and then a specially designed "back substitution" procedure is harnessed to correct the predictor. Its numerical performance was reported to outperform some very efficient algorithms such as HTY method. As the Jacobi fashion is preferable provided the parallel computing infrastructure is available, Han et. al proposed an augmented Lagrangian based parallel splitting (ALBPS) method in which all the subproblems can be computed simultaneously [12]. Its correction step is just a simple extension on all the working variables, so its extra computational cost is low. He et. al also proposed an algorithm similar to that in [14], but the lower bound of extension step size was refined. Combine the idea of correcting the output of ADMM and updating the multiplier multiple times, Dai et. al proposed a sequential updating scheme of the

Lagrange multiplier (SUSLM) [7] in which the predictor is generated by the following procedure:

$$\begin{cases} \lambda^{k+0} = \lambda^k - \beta((\sum_{i=1}^p A_i x_i^k) - c). \\ \tilde{x}_1^k = \arg\min\{f_1(x_1) - \langle \lambda^{k+0}, Ax_1\rangle + \frac{\mu\beta}{2}\|A_1(x_1 - x_1^k)\|^2 \mid x_1 \in \mathcal{X}_1\}, \\ ..., \\ \lambda^{k+\frac{i}{p}} = \lambda^{k+\frac{i-1}{p}} - \mu\beta A_{i-1}(\tilde{x}_{i-1}^k - x_{i-1}^k). \\ \tilde{x}_i^k = \arg\min\{f_i(x_i) - \langle \lambda^{k+\frac{i}{p}}, Ax_i\rangle + \frac{\mu\beta}{2}\|A_i(x_i - x_i^k)\|^2 \mid x_i \in \mathcal{X}_i\}, \\ ..., \\ \tilde{\lambda}^k = \lambda^k - \frac{\beta}{2}((\sum_{i=1}^p A_i \tilde{x}_i^k) - c), \end{cases} \tag{1.8}$$

with $\mu \geq 1$, then a correction step is followed to guarantee the convergence. Specially, its predicting stage (1.8) reduces to the standard ADMM scheme provided $\mu = 1$, hence SUSLM can be seen as a generalization of ADMMG. However, the parameter condition of SUSLM is still conservative which may restrict its numerical performance. We would like to derive an improved SUSLM algorithm with relaxed step sizes, and the proposed algorithm is expected to have better numerical behaviour than the original SUSLM as well as other efficient competitors.

The rest part of the present paper is organized as follows. In the second section, we define some necessary notations and give some preliminaries for the theoretical analysis. We present the new algorithm as well as its convergence properties in Section 3. Then the preliminary experimental results are reported in the fourth section. Finally, some conclusions are given in Section 5.

2 Preliminaries

In this section, some useful notations are defined, and some basic concepts are given for later analysis.

Let $x = ({x_1}^\top, ..., {x_p}^\top)^\top$, $u = (x^\top, \lambda^\top)^\top$, $\Omega = \mathcal{X}_1 \times ... \times \mathcal{X}_p \times \mathcal{R}^n$, $\|\cdot\|$ denote the Euclidean norm $\|\cdot\|_2$, $\|u\|_H := \sqrt{u^\top H u}$ be a norm with given positive definite matrix H and I represent an identity matrix with appropriate dimension.

As our theoretical analysis is based on the variational inequalities (VI) of (1.1), according to the first-order optimality condition, a VI reformulation to characterize the solution of (1.1) is as follows: finding $u^* = ({x_1^*}^\top, ..., {x_p^*}^\top, {\lambda^*}^\top)^\top \in \Omega$ such that

$$\begin{cases} x_i^* \in \mathcal{X}_i,\ f_i(x_i) - f_i(x_i^*) + (x_i - x_i^*)^\top(-A_i^\top\lambda^*) \geq 0,\ \forall x_i \in \mathcal{X}_i,\ i = 1, ..., p, \\ \lambda^* \in \mathcal{R}^n, \quad (\lambda - \lambda^*)^\top((\sum_{i=1}^p A_i x_i^*) - c) \geq 0,\ \forall \lambda \in \mathcal{R}^n. \end{cases} \tag{2.1}$$

Furthermore, the variational inequalities in (2.1) can be rewritten as the following compact form:

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

with

$$f(x) := \sum_{i=1}^p f_i(x_i), F(u) := \begin{pmatrix} -A_1^\top\lambda \\ \vdots \\ -A_p^\top\lambda \\ Ax - c \end{pmatrix}. \tag{2.3}$$

Due to the skew-symmetricity of the mapping $F(\cdot)$, $(\tilde{u} - \hat{u})^\top(F(\tilde{u}) - F(\hat{u})) = 0$ holds for any $\tilde{u}, \hat{u} \in \Omega$, hence $F(\cdot)$ is monotone. For simplicity, we denote (2.2) by VI(Ω,F,f) and its solutions set by Ω^* throughout this paper.

For the convenience of future theoretical analysis, we introduce the matrices M, N and Q defined as follows:

$$M := \begin{pmatrix} \frac{\mu\beta}{\gamma_x}A_1^\top A_1 & 0 & \cdots & 0 & 0 \\ \mu\beta A_2^\top A_1 & \frac{\mu\beta}{\gamma_x}A_2^\top A_2 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mu\beta A_p^\top A_1 & \mu\beta A_p^\top A_2 & \cdots & \frac{\mu\beta}{\gamma_x}A_p^\top A_p & 0 \\ 0 & 0 & \cdots & 0 & \frac{1}{\beta\gamma_\lambda}I \end{pmatrix}, \tag{2.4}$$

$$N := \begin{pmatrix} \frac{2\mu\beta}{\gamma_x}A_1^\top A_1 & \mu\beta A_1^\top A_2 & \cdots & \mu\beta A_1^\top A_p & A_1^\top \\ \mu\beta A_2^\top A_1 & \frac{2\mu\beta}{\gamma_x}A_2^\top A_2 & \cdots & \mu\beta A_2^\top A_p & A_2^\top \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mu\beta A_p^\top A_1 & \mu\beta A_p^\top A_2 & \cdots & \frac{2\mu\beta}{\gamma_x}A_p^\top A_p & A_p^\top \\ A_1 & A_2 & \cdots & A_p & \frac{2}{\beta\gamma_\lambda}I \end{pmatrix} \tag{2.5}$$

and

$$Q := \begin{pmatrix} 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 \\ A_1 & A_2 & \cdots & A_p & 0 \end{pmatrix}. \tag{2.6}$$

It is easy to see that the matrices defined above satisfy the relationship:

$$N = M + M^\top + Q + Q^\top. \tag{2.7}$$

3 A new sequential updating scheme of the Lagrange multiplier with relaxed step sizes

In this section, we first describe the iterative scheme of our new algorithm, and then its convergence results is established in the framework of variational inequality.

3.1 Algorithm

Our new algorithm is as follows:

Algorithm 1 (A new sequential updating scheme of the Lagrange multiplier with relaxed step sizes (SUSLMR) for (1.1)).
Set $\mu \geq 0$, $\gamma_x, \gamma \in (0,2)$, $\gamma_\lambda \in (0, 2\mu)$, $\beta > 0$ and $G \in \mathcal{S}_{++}$ properly. With given k-th iterate $u^k = ({x^k}^\top, {\lambda^k}^\top)^\top$, the next iterate u^{k+1} is computed via the following steps:

$$\begin{cases} \lambda^{k+0} = \lambda^k - \beta((\sum_{i=1}^p A_i x_i^k) - c). \\ \tilde{x}_1^k = \arg\min\{f_1(x_1) - \langle \lambda^{k+0}, Ax_1\rangle + \frac{\mu\beta}{2\gamma_x}\|A_1(x_1 - x_1^k)\|^2 \mid x_1 \in \mathcal{X}_1\}, \\ ..., \\ \lambda^{k+\frac{i}{p}} = \lambda^{k+\frac{i-1}{p}} - \mu\beta A_{i-1}(\tilde{x}_{i-1}^k - x_{i-1}^k). \\ \tilde{x}_i^k = \arg\min\{f_i(x_i) - \langle \lambda^{k+\frac{i}{p}}, Ax_i\rangle + \frac{\mu\beta}{2\gamma_x}\|A_i(x_i - x_i^k)\|^2 \mid x_i \in \mathcal{X}_i\}, \\ ..., \\ \tilde{\lambda}^k = \lambda^k - \frac{\gamma_\lambda\beta}{2}((\sum_{i=1}^p A_i\tilde{x}_i^k) - c), \\ \alpha^k = \|u^k - \tilde{u}^k\|_N^2/(2\|G^{-1}M(u^k - \tilde{u}^k)\|_G^2), \\ u^{k+1} = u^k - \gamma\alpha^k G^{-1}M(u^k - \tilde{u}^k), \\ k = k + 1. \end{cases} \tag{3.1}$$

Remark 3.1 *The superscript "k+0" is slightly abused and it is different with superscript "k".*

Remark 3.2 *In the correction step, the step size α^k in the correction step is an "optimal" step size from the aspect of contractive properties.*

Remark 3.3 *In the proposed algorithm, the parameters γ_x and γ_λ are introduced to accelerate the convergence. When they are both fixed to be 1, the new algorithm reduces to its ancestor SUSLM, hence it can be seen as a generalization of SUSLM. In the new algorithm, the interval of parameter μ is extended from $[1, +\infty)$ to $(0, \infty)$, the interval of parameter γ_x is extended from $\{1\}$ to $(0, 2)$, and the interval of parameter γ_λ is extended from $\{1\}$ to $(0, 2\mu)$. Therefore, we have more freedom in choosing the parameter settings which potentially enables better numerical behaviour.*

3.2 Convergence analysis

We now turn to establish the convergence results of the proposed algorithm under mild assumptions.

We first show how to examine the optimality of the current iterate at each iteration.

Lemma 3.1 *Let $\tilde{u}^k$ be generated by (3.1) with given u^k. Then $\tilde{u}^k$ is a solution of VI(Ω,F,f) if $A_j\tilde{x}_j^k = A_j x_j^k$ $(i = 1, ..., p)$ and $\tilde{\lambda}^k = \lambda^k$ hold.*

Proof: We first write the optimality condition of the x_i-subproblem in (3.1) as: Find $\tilde{x}_i^k \in \mathcal{X}_i$ such that

$$f_i(x_i) - f_i(\tilde{x}_i^k) + (x_i - \tilde{x}_i^k)^\top\{-A_i^\top\lambda^{k+\frac{i}{p}} + \frac{\mu\beta}{\gamma_x}A_i^\top A_i(\tilde{x}_i^k - x_i^k)\} \geq 0,\ \forall x_i \in \mathcal{X}_i. \tag{3.2}$$

Invoking the definitions of $\lambda^{k+\frac{i}{p}}$ and $\tilde{\lambda}^k$ in (3.1), we rewrite the above condition as follows:

$$\begin{aligned} &f_i(x_i) - f_i(\tilde{x}_i^k) + (x_i - \tilde{x}_i^k)^\top\{-A_i^\top\tilde{\lambda}^k - \beta A_i^\top\sum_{j=1}^{p}A_j(\tilde{x}_j^k - x_j^k) \\ &+\frac{\mu\beta}{A_{\ i}}^\top\sum_{j=1}^{i-1}A_j(\tilde{x}_j^k - x_j^k) + \frac{\mu\beta}{\gamma_x}A_i^\top A_i(\tilde{x}_i^k - x_i^k)\} \geq 0,\ \forall x_i \in \mathcal{X}_i. \end{aligned}$$

Noting $\tilde{\lambda}^k = \lambda^k - \beta((\sum_{i=1}^{p}A_i\tilde{x}_i^k) - c)$ and taking the sum of (3.3) from $i = 1$ to p, we obtain

$$\begin{aligned} f(x) - f(\tilde{x}^k) + &\begin{pmatrix} x_1 - \tilde{x}_1^k \\ \vdots \\ x_p - \tilde{x}_p^k \\ \lambda - \tilde{\lambda}^k \end{pmatrix}^\top \left\{ \begin{pmatrix} -A_1^\top\tilde{\lambda}^k \\ \cdots \\ -A_p^\top\tilde{\lambda}^k \\ A\tilde{x}^k - c \end{pmatrix} - \beta\begin{pmatrix} A_1^\top \\ \vdots \\ A_p^\top \\ 0 \end{pmatrix}\sum_{j=1}^{p}A_j(\tilde{x}_j^k - x_j^k) \right. \\ &\left. + \begin{pmatrix} \frac{\mu\beta}{\gamma_x}A_1^\top A_1 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mu\beta A_p^\top A_1 & \mu\beta A_p^\top A_2 & \cdots & \frac{\mu\beta}{\gamma_x}A_p^\top A_p & 0 \\ 0 & 0 & \cdots & 0 & \frac{1}{\beta\gamma_\lambda}I \end{pmatrix}\begin{pmatrix} \tilde{x}_1^k - x_1^k \\ \vdots \\ \tilde{x}_p^k - x_p^k \\ \tilde{\lambda}^k - \lambda^k \end{pmatrix} \right\} \geq 0,\ \forall u \in \Omega. \end{aligned} \tag{3.3}$$

To simplify the notation, we define the function $\phi(u^k, u^k)$ as

$$\phi(u^k, \tilde{u}^k) := \beta\begin{pmatrix} A_1^\top \\ \vdots \\ A_p^\top \\ 0 \end{pmatrix}\sum_{j=1}^{p}A_j(\tilde{x}_j^k - x_j^k). \tag{3.4}$$

Combining the definitions of F and M in (2.3) and (2.4) respectively, the relation (3.3) can be written in a more compact way:

$$f(x) - f(\tilde{x}^k) + (u - \tilde{u}^k)^\top\{F(\tilde{u}^k) - \phi(u^k, \tilde{u}^k) + M(\tilde{u}^k - u^k)\} \geq 0,\ \forall u \in \Omega. \tag{3.5}$$

It is easy to see that when $A_j(\tilde{x}_j^k - x_j^k) = 0$ $(i = 1, ..., p)$ and $\tilde{\lambda}^k - \lambda^k = 0$ hold, the last two terms in the curly bracket of (3.5) will vanish, hence the optimality of $\tilde{u}^k$ is arrived.

In the next lemma, $M(u^k - \tilde{u}^k)$ is shown to be an ascent direction of the merit function $\|u^k - u^*\|^2$ at u^k.

Lemma 3.2 *Let $u^* = \binom{x^*}{\lambda^*}$ be an arbitrary solution of (1.1). Then for any u^k, the predictor $\tilde{u}^k$ generated by Algorithm 1 satisfies*

$$(u^k - u^*)^\top M(u^k - \tilde{u}^k) \geq \frac{1}{2}\|u^k - \tilde{u}^k\|_N^2. \tag{3.6}$$

Proof: Combining (2.2) with $u = \tilde{u}^k$ and (3.5) with $u = u^*$, we arrive at

$$(\tilde{u}^k - u^*)^\top\{F(u^*) - F(\tilde{u}^k) + \phi(u^k, \tilde{u}^k) - M(\tilde{u}^k - u^k)\} \geq 0. \tag{3.7}$$

Noting the monotonicity of $F(u)$ and , we rewrite (3.7) as

$$\begin{aligned}(\tilde{u}^k-u^*)^\top M(u^k-\tilde{u}^k) &\geq (u^*-\tilde{u}^k)^\top \phi(u^k,\tilde{u}^k)\\ &= \beta(Au^*-A\tilde{u}^k)^\top \sum_{j=1}^{p} A_j(\tilde{x}_j^k-x_j^k)\\ &= (\tilde{\lambda}^k-\lambda^k)^\top \sum_{j=1}^{p} A_j(\tilde{x}_j^k-x_j^k)\\ &= (\tilde{u}^k-u^k)^\top Q(\tilde{u}^k-u^k).\end{aligned} \tag{3.8}$$

Adding the term $(u^k-\tilde{u}^k)^\top M(u^k-\tilde{u}^k)$ to both sides of (3.8), we finally obtain

$$\begin{aligned}(u^k-u^*)^\top M(u^k-\tilde{u}^k) &\geq (\tilde{u}^k-u^k)^\top (M+Q)(\tilde{u}^k-u^k)\\ &= \frac{1}{2}(\tilde{u}^k-u^k)^\top (M+M^\top+Q+Q^\top)(\tilde{u}^k-u^k)\\ &= \frac{1}{2}\|u^k-\tilde{u}^k\|_N^2,\end{aligned} \tag{3.9}$$

which is the assertion to prove.

Recalling that the submatrices A_i are all full column rank, noting $\gamma_x \in (0,2)$ and $\gamma_\lambda \in (0,2\mu)$, then the matrix N as defined in (2.5) is positive definite. Invoking the definition of α^k in Algorithm 1, we immediately derive the lower bound of $\{\alpha_k\}$ as follows:

$$\alpha^k = \frac{\|u^k-\tilde{u}^k\|_N^2}{2\|G^{-1}M(u^k-\tilde{u}^k)\|_G^2} \geq \frac{\mathrm{eig}_{\min}(N)}{2\|M^\top G^{-1}M\|} := \underline{\alpha} > 0. \tag{3.10}$$

Based on the above results, convergence result of Algorithm 1 can be established.

Theorem 3.1 *Let $\{u^k\}$ be the sequence generated by Algorithm 1, then it converges to a solution of VI(Ω,F,f).*

Proof: It follows from the correction step in Algorithm 1 that:

$$\begin{aligned}\|u^{k+1}-u^*\|_G^2 =& \|u^k-u^*\|_G^2 - 2\gamma\alpha^k(u^k-u^*)M(u^k-\tilde{u}^k)\\ &+\gamma^2{\alpha^k}^2\|G^{-1}M(u^k-\tilde{u}^k)\|_G^2\\ \leq& \|u^k-u^*\|_G^2 - \frac{\gamma(2-\gamma)}{2}\alpha^k\|u^k-\tilde{u}^k\|_N^2\\ \leq& \|u^k-u^*\|_G^2 - \frac{\gamma(2-\gamma)}{2}\underline{\alpha}\|u^k-\tilde{u}^k\|_N^2,\end{aligned} \tag{3.11}$$

where first inequality is from (3.6) and the second one comes from (3.10). Hence $\{\|u^k-u^*\|_G^2\}$ is monotonically nonincreasing (thus $\{u^k\}$ is bounded), and $\|u^k-\tilde{u}^k\|_N^2 \to 0$ as $k\to\infty$. We have subsequently $\|x_i^k-\tilde{x}_i^k\| \to 0$ $(i=1,2,...)$ and $\|\lambda^k-\tilde{\lambda}^k\| \to 0$ due to the positive definiteness of N. According to the result of Lemma 3.1, the sequence $\{\tilde{u}^k\}$ converges to a solution of VI(Ω,F,f). Due to $\|u^k-\tilde{u}^k\|_N^2 \to 0$ as $k\to\infty$, the sequence $\{u^k\}$ also converges to this solution. The assertion is proved.

4 Numerical Experiments

In this section, numerical experiments on the proposed algorithm are conducted to verify its efficiency for solving two types of problems. All the algorithms were coded in MATLAB R2015a by the present authors and carried out on a desktop computer with Intel Core i7 CPU at 3.6GHz, 8 GB memory and Windows 7 OS.

4.1 Numerical results on LCQP problem

We first consider the following linearly constrained quadratic programming (LCQP):

$$\begin{aligned}\min_x \quad & f_1(x_1)+...+f_p(x_p)\\ \text{s.t.} \quad & A_1x_1+...+A_px_p = c,\end{aligned} \tag{4.1}$$

where $f_i(x_i) = \frac{1}{2}x_i^\top H_i x_i + x_i^\top q_i (i = 1, ..., p) \in \mathcal{R}^{m_i} \to \mathcal{R}$ with $H_i \in \mathcal{R}^{m_i \times m_i}$, $q \in \mathcal{R}^{m_i}$, $A_i \in \mathcal{R}^{n \times m_i}$ and $c \in \mathcal{R}^n$. In order to simplify the experiments, we let $m_1 = ... = m_p := m$ in this subsection.

To reveal the efficiency of the new algorithm, we compare our algorithm (denoted by "SUSLMR") with its ancestor SUSLM and two efficient ADMM-based algorithms: the Block-wise ADMM with Relaxation factor [20] (denoted by "BADMMR"); the Generalized Symmetric ADMM [1] (denoted by "GSADMM").

In order to implement BADMMR and GSADMM, we first need to regroup the variables of (4.1) into two groups. In fact, the scheme of these two algorithms are both mixture of Jacobi and Gauss-Seidel and we have freedom to choose the grouping strategy. For instance, when $p = 3$, we can regroup the 3 blocks variables by either 1∼2 or 2∼1 strategy which can lead to two iterative schemes for these two algorithms respectively, see [20, 1] for details.

Note that all the splitting subproblems for solving (4.1) are unconstrained quadratic programming which is equivalent to linear equations. Since the Hessian (coefficient matrix) in all the subproblem are fixed, we can speed up the algorithm by doing Cholesky decomposition on these Hessian at the beginning of the algorithm, then the computation of each subproblem can be decomposed into two simpler subproblems whose coefficient matrices are lower and upper triangle matrices respectively.

The test problems are constructed by the following procedure:

- the matrix A_i $(i = 1, ..., p)$ is randomly generated such that all its entries are i.i.d. Gaussian, i.e., $A_i = randn(n, m)$;
- the symmetric positive definite matrices H_i $(i = 1, ..., p)$ are generated by:
  a) first generate a matrix $\tilde{H}_i$ whose entries are i.i.d. Gaussian, i.e. $\tilde{H}_i = randn(m)$
  b) then H_i is obtained by $H_i = \tilde{H}_i^\top * \tilde{H}_i$;
- then the optimal solution is generated by $x_i^* = randn(m, 1)$ $(i = 1, ..., p)$ and $\lambda^* = randn(n, 1)$;
- we finally obtain q_i $(i = 1, ..., p)$ and c by: $q_i = -H_i * x_i^* + A_i^\top \lambda^*$ and $c = \sum_{i=1}^p A_i * x_i^*$ such that $u^* = (x_1^{*\top}, ..., x_p^{*\top}, \lambda^{*\top})^\top$ satisfies the KKT condition.

Since the optimal solution is known when the problem is generated, we are able to evaluate the accuracy of the current iterate by the distance to the solution defined as follows:

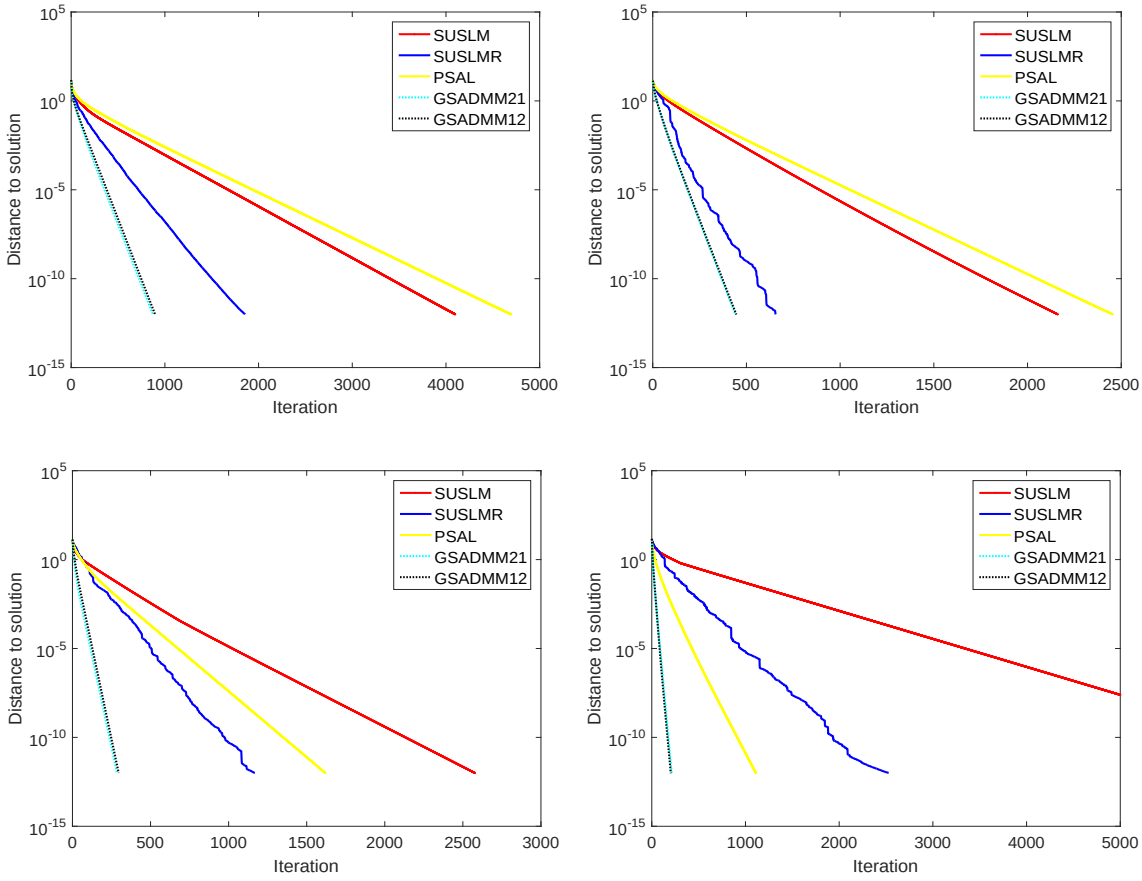
$$\text{dis}(u^k) := \max\{\|x_1^k - x_1^*\|, ..., \|x_p^k - x_p^*\|, \|\lambda^k - \lambda^*\|\}. \tag{4.2}$$

We initialize all working variables with zero vectors for all the test algorithms, and stop them when either a maximal number of iterations (denoted by "maxit") is attained or the distance to the optimal solution of all working variables defined as (4.2) are smaller than a prescribed tolerance (denoted by "tol"), i.e., $\text{dis}(k) < \text{tol}$. Unless otherwise specified, the default setting of maxit and tol are 5000 and 10^{-12}, respectively.

The setting of parameter β in all the test algorithms is critical to their performances, but its optimal setting can be different with different problem settings, hence its setting is empirically chosen to maximize their performances in our experiments. We usually take $\beta = 10^{-2}$ or $\beta = 10^{-1}$ in our experiments for small-scale problems. The settings of newly introduced parameters γ_x and γ_λ in our algorithm are also critical to its performance. Our initial guess is to take their values close to their upper bound, e.g., $\gamma_x = 1.9$, while numerical evidences show that their optimal settings still depend on the problem setting. We take $\gamma_x = 0.7$ and $\gamma_\lambda = 1.9$ under most settings, and we observe that the setting of γ_x is empirically more important than that of γ_λ. The parameter γ in the correction step of our algorithm and its ancestor SUSLM is usually chosen to be between 1 and 2, and we usually take $\gamma = 1.2$ in our experiments. Another important parameter μ in our algorithm and SUSLM is fixed to be 1 in our experiments as suggested in [7]. In the iterating scheme of GSADMM which requires two extension factors on the two updates of multipliers respectively, we set them to be 0.9 and 1.09 as suggested in [1]. Note the GSADMM can take different grouping strategies as the variables need to be regrouped into two groups, for simplicity, the grouping strategy is fixed to be $p/2 \sim p/2$ provided p is an even number.

To investigate the convergence behavior of the test algorithms, we tested four cases: 3-block, 6-block, 10-block and 20-block, and the experimental results are plotted in Figure 1, 2, 3 and 4, respectively. For each case, n is fixed to be 100, and m takes four different settings. We can observe that the convergence of our algorithm is always faster than that of its ancestor SUSLM which means the introduction of two parameters

Figure 1: Experimental results on 3-block problem with (n, m_i)=(100,50), (100,60), (100,70) and (100,80), respectively.



is critical to its performance. In addition, the convergence of our algorithm is faster than its competitors such as HTY or GSADMM under a part of problem settings, which means our algorithm is an overall competitive algorithm for solving the LCQP problem in the form of (4.1).

4.2 Numerical results on RPCA problem

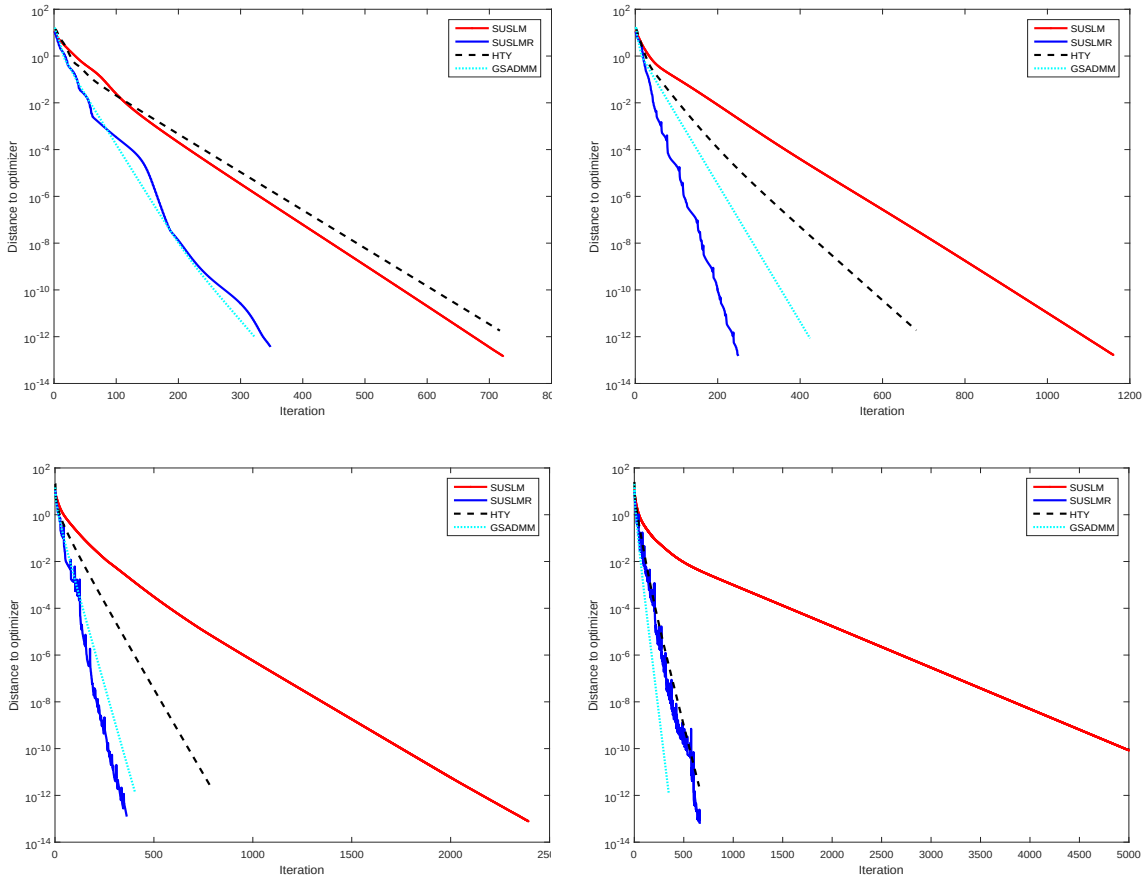
In this subsection, we test the robust principal component analysis (RPCA) problem which arises from various areas such as model selection or image processing, see [4, 31]. This problem aims at recovering a low-rank and a sparse matrices from their sum. Nonconvex regularization has been introduced, but the resulting optimization problem is nonconvex and hence intractable. Specifically, the following convex surrogate problem is usually adopted:

$$\begin{aligned} \min_{A,E} \quad & \|A\|_* + \mu\|E\|_1 \\ \text{s.t.} \quad & A + E = C, \end{aligned} \tag{4.3}$$

where $C \in \mathcal{R}^{m\times n}$ is the data matrix. The nuclear norm $\|\cdot\|_*$ is a convex surrogate which is to induce the low-rank component of C while the elementwise ℓ_1 norm $\|\cdot\|_1$ can catch the sparse component of C. It has been verified that, under certain mild assumptions, the model (4.3) can recover the original solution accurately. Readers can refer to [4] for detailed introduction of RPCA.

In many real applications, the observation could be corrupted by additive Gaussian noise, hence the equality

Figure 2: Experimental results on 6-block problem with (n, m_i)=(100,40), (100,50), (100,60) and (100,70), respectively.



constraint in (4.3) may not be reasonable. To this end, the following relaxation model was suggested [30]:

$$\begin{aligned} \min_{A,E} \quad & \|A\|_* + \mu\|E\|_1 \\ \text{s.t.} \quad & A + E + Z = C, \quad \|Z\|_F \le \delta, \end{aligned} \tag{4.4}$$

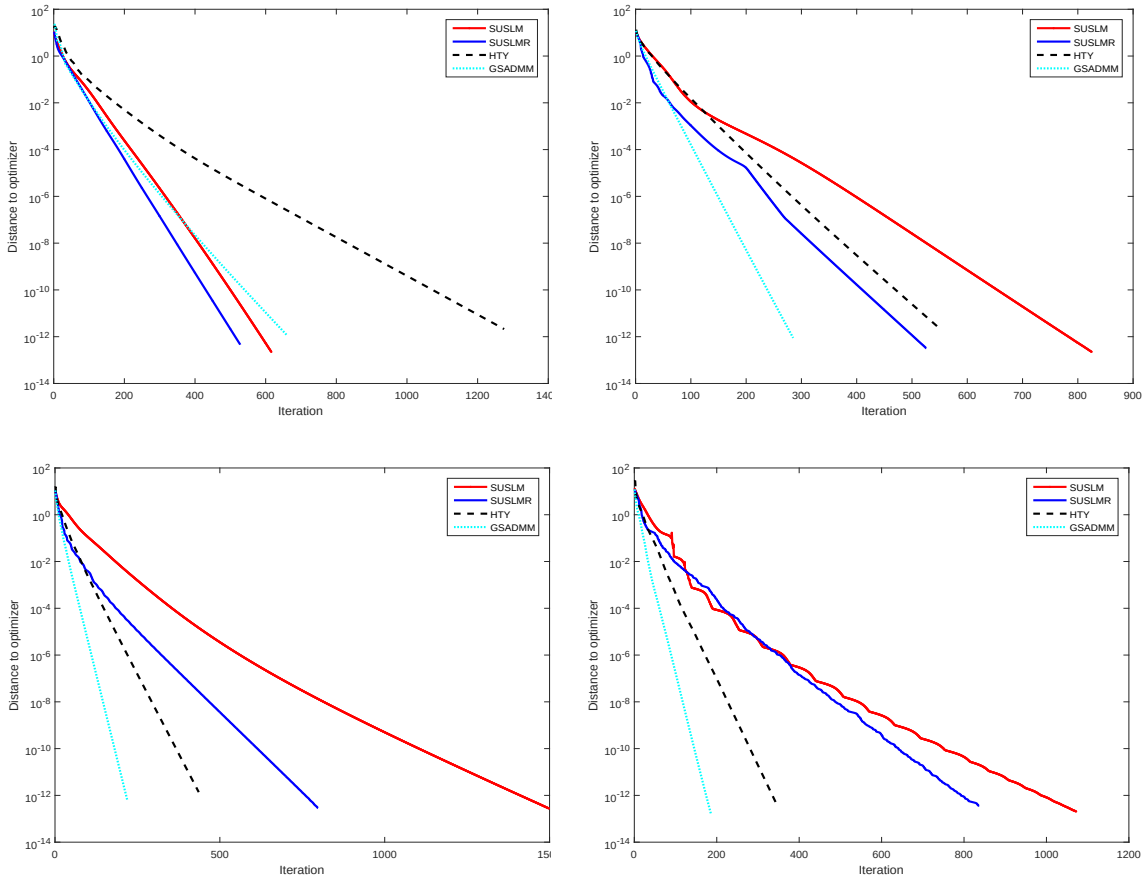
where $\|\cdot\|_F$ represents the Frobenius norm and δ is a parameter related to the noise level.

We observe that model (4.4) is a special case of the general multi-block model (1.1) with 3 block variables, hence our algorithm should be suitable for solving it. Moreover, taking advantage of the special structures of nuclear norm $\|A\|_*$ and ℓ_1 norm $\|E\|_1$, the splitting subproblems involved in the test algorithms are simple enough to have closed form solution: the A-subproblem is solved by a partial singular value decomposition (SVD); the E-subproblem is solved by a soft shrinkage operation; and the Z-subproblem is solved by a projection onto a ball. Although the cost at each iteration is dominated by a partial SVD, and there have been a bunch of efficient solvers for partial SVD (e.g., LMSVD, SLRP, ...), however, as reducing the per-iteration cost is not our main concern, we implement the classical PROPACK solver to compute the partial SVD for all the test algorithms.

To reveal the performance of our algorithm, the proposed algorithm is compared with some efficient algorithms: Generalized Symmetric ADMM [1] in $1 \sim 2$ or $2 \sim 1$ fashion (denoted by GSADMM12 and GSADMM21, respectively); the splitting method [19] proposed by He, Tao and Yuan (denoted by HTY); the partial splitting augmented Lagrangian method [10] proposed by Han and et. al (denoted by PSAL).

We compare the test algorithms from the aspects of convergence speed and solution accuracy. To measure

Figure 3: Experimental results on 10-block problem with (n, m_i)=(100,20), (100,30), (100,40) and (100,50), respectively.



the solution accuracy, we use the relative error of A and E with the accurate solutions defined as follows:

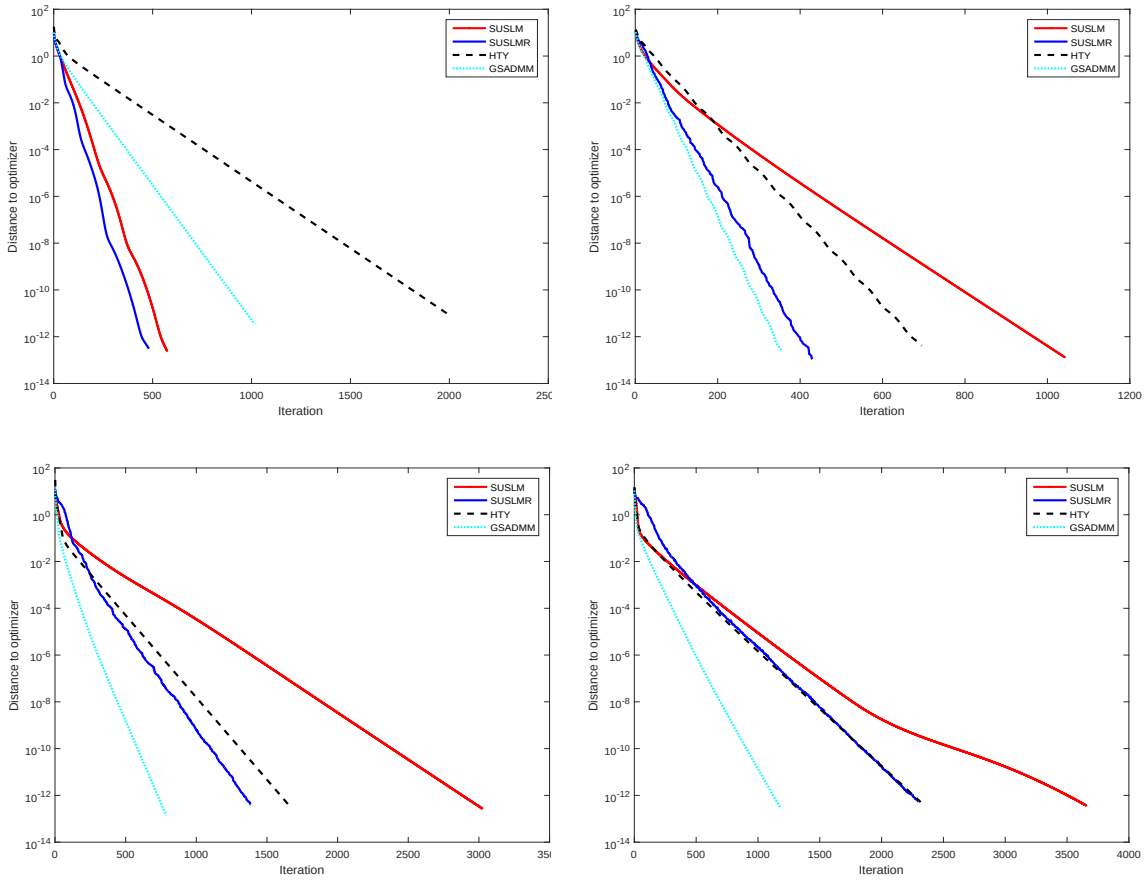
$$\text{err}(A) := \frac{\|A^k - A^*\|_F}{\|A^*\|_F} \quad \text{and} \quad \text{err}(E) := \frac{\|E^k - E^*\|_F}{\|E^*\|_F}. \tag{4.5}$$

The data matrix A is constructed by $A = UV^\top$ where $U \in \mathcal{R}^{m\times k}$ and $V \in \mathcal{R}^{n\times k}$ are randomly constructed such that all their entries are i.i.d. Gaussian. The sparse matrix E is randomly generated whose non-zero entries are i.i.d. uniformly distributed in the interval [-50, 50], and SR denotes the ratio of non-zero entries, i.e., $\|E\|_0/mn$. The noise matrix Z is constructed by two steps: randomly generated a matrix Z such that its entries are i.i.d. Gaussian, then it is scaled such that $\|Z\|_F = \eta\|A + E\|_F$ where η is a parameter related to the noise level. The default settings of necessary parameters are as follows: $\mu = 3$, $\eta = 0.01$, $\delta = 0.001$. The settings of other parameters in the test algorithms follow that in the previous experiments.

All the variables in the test algorithms were initialized with random matrices whose entries are i.i.d. Gaussian. Due to the presence of Gaussian noise in the observation C, accurate solution is not attainable and thus too many iterations could be meaningless. Instead, we run all the test algorithms for fixed 150 iterations.

The experimental results under two different problem settings are shown in Figure 5 and 6, respectively. The performance of each algorithm can be different with different problem settings. We observe from Figure 5 that: when $k = 5$, $SR = 5\%$, PSAL converges faster than the other algorithms, and it can achieve the best accuracy, but the performance gap between our algorithm and PSAL is small; when $k = 5$, $SR = 50\%$, our algorithm can obtain the most accurate solution while GSADMM12 converges faster at the beginning stage of iteration progress. We then observe Figure 6 and find that, when $k = 20$, our algorithm is always the best one among all the test algorithms from the aspects of both convergence speed and solution accuracy.

Figure 4: Experimental results on 20-block problem with (n, m_i)=(100,8), (100,10), (100,20) and (100,30), respectively.



The above experimental results indicate that the proposed algorithm is effective for solving RPCA problem, and its performance is competitive compared with some most efficient algorithms.

5 Conclusions

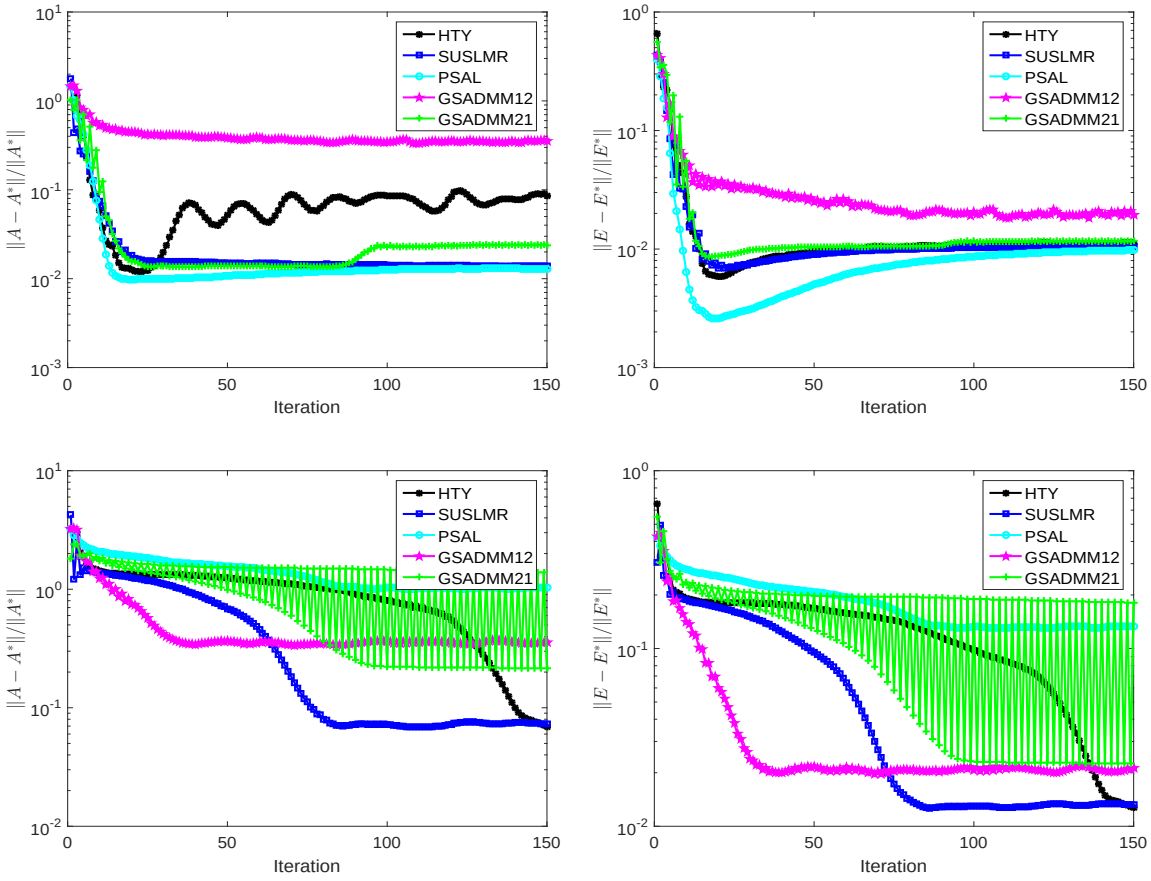
Due to a wide range of applications, the multi-block linearly constrained convex optimization has been extensively studied in the past few years. Based on the sequential updating scheme of the Lagrange multiplier proposed by Dai and et.al, a new SUSLM with relaxed step sizes is proposed in this paper. In the new scheme, two parameters are introduced and the step size condition is relaxed. We derive the contractive properties of the proposed algorithm from the perspective of variational inequalities. Its iterating scheme is similar to that of original SUSLM, hence it is easy to implement. A synthetic LCQP problem and a practical RPCA problem have been tested, and its numerical efficiency is then justified.

In future, by plugging some more strategies and techniques such as indefinite proximal term strategy or self-adoptive step size rule into the scheme, its numerical performance is expected to be even better.

References

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

Figure 5: Iteration progress of relative error, $(m, n, k) = (100, 100, 5)$. First row: $SR = 5\%$; Second row: $SR = 50\%$; .



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

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

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

[5] C.H. Chen, B.S. He, Y.Y. Ye, 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.

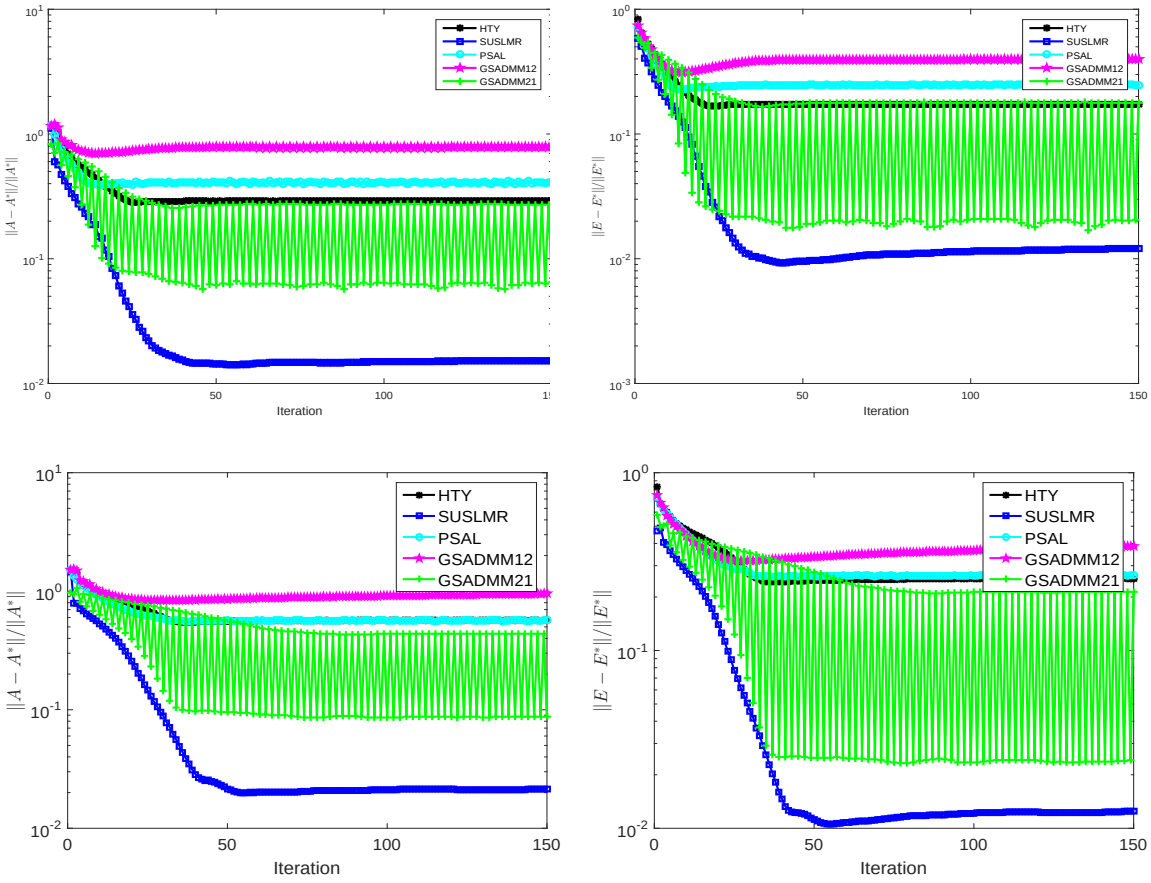
[6] C.H. Chen, Y. Shen, and Y.F. You. On the convergence analysis of the alternating direction method of multipliers with three blocks. *Abstr. Appl. Anal.*, 2013, Article ID 183961:7 pages, 2013.

[7] Y.H. Dai, D.R. Han, and W.X. Zhang. A sequential updating scheme of the lagrange multiplier for separable convex programming. *Math. Comput.*, 86:315–343, 2017.

[8] D. Gabay and B. Mercier. A dual algorithm for the solution of nonlinear variational problems via finite element approximations. *Comput. Math. Appl.*, 2:17–40, 1976.

[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.

Figure 6: Iteration progress of relative error, $(m, n, k) = (100, 100, 20)$. First row: $SR = 5\%$; Second row: $SR = 10\%$; .



[10] D.R. Han, W.W. 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.

[11] 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.

[12] D.R. Han, X.M. 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.

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

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

[15] B.S. He, L. Liao, D. Han, and H. Yang. A new inexact alternating directions method for monotone variational inequalities. *Math. Program.*, 92(1, Ser. A):103–118, 2002.

[16] B.S. He, H. Liu, Z.R. Wang, and X.M. Yuan. A strictly contractive peaceman-rachford splitting method for convex programming. *SIAM J. Optim.*, 24(3):1011–1040, 2014.

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

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

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

[20] B.S. He, M.H. 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] 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.

[22] B.S. He 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.

[23] H. He, B.and Liu, J.W. Lu, and X.M. Yuan. *Application of the Strictly Contractive Peaceman-Rachford Splitting Method to Multi-Block Separable Convex Programming*, pages 195–235. Splitting Methods in Communication, Imaging, Science, and Engineering. Scientific Computation. Springer, Cham, 2016.

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

[25] 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.

[26] L.S. Hou, H.J. 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.

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

[28] Y. Shen, Z.W. 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.

[29] Y. Shen, X.Y. Zhang, and X.Y. Zhang. A partial ppa block-wise admm for multi-block linearly constrained separable convex optimization. Manuscript, 2018. Submitted to *Optimization*.

[30] 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.

[31] 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.

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