Mathematics of Operations Research

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Convergence Rate Analysis for the Alternating Direction Method of Multipliers with a Substitution Procedure for Separable Convex Programming

Bingsheng He, Min Tao, Xiaoming Yuan

To cite this article:
Bingsheng He, Min Tao, Xiaoming Yuan (2017) Convergence Rate Analysis for the Alternating Direction Method of Multipliers with a Substitution Procedure for Separable Convex Programming. Mathematics of Operations Research 42(3):662-691.
https://doi.org/10.1287/moor.2016.0822

http://pubsonline.informs.org/journal/moor/

MATHEMATICS OF OPERATIONS RESEARCH
Vol. 42, No. 3, August 2017, pp. 662–691
ISSN 0364-765X (print), ISSN 1526-5471 (online)

Convergence Rate Analysis for the Alternating Direction Method of Multipliers with a Substitution Procedure for Separable Convex Programming

Bingsheng He,[a,b] **Min Tao,**[c] **Xiaoming Yuan**[d]

[a] Department of Mathematics, South University of Science and Technology of China, Shenzhen 518055, China; [b] Department of Mathematics, Nanjing University, Nanjing 210093, China; [c] Department of Mathematics, Nanjing University, Nanjing, 210093, China; [d] Department of Mathematics, Hong Kong Baptist University, Hong Kong, China

Contact: hebma@nju.edu.cn (BH); taom@nju.edu.cn (MT); xmyuan@hkbu.edu.hk (XY)

Received: September 18, 2012
Revised: December 12, 2013, January 12, 2015, April 6, 2016, and June 12, 2016
Accepted: August 4, 2016
Published Online in Articles in Advance: February 7, 2017

MSC2010 Subject Classification: 90C25; 90C30
OR/MS Subject Classification: Nonlinear programming

https://doi.org/10.1287/moor.2016.0822

Abstract. Recently, in He et al. [He BS, Tao M, Yuan XM (2012) Alternating direction method with Gaussian back substitution for separable convex programming. *SIAM J. Optim.* 22(2):313–340], we have showed the first possibility of combining the Douglas-Rachford alternating direction method of multipliers (ADMM) with a Gaussian back substitution procedure for solving a convex minimization model with a general separable structure. This paper is a further study on this theme. We first derive a general algorithmic framework to combine ADMM with either a forward or backward substitution procedure. Then, we show that convergence of this framework can be easily proved from the contraction perspective, and its local linear convergence rate is provable if certain error bound condition is assumed. Without such an error bound assumption, we can estimate its worst-case convergence rate measured by the iteration complexity.

Funding: Bingsheng He was supported by the National Natural Science Foundation of China (NSFC) [Grant 11471156]. Min Tao was supported by the NSFC [Grant 11301280] and the Fundamental Research Fund for the Central Universities [Grant 020314330019]. Xiaoming Yuan was supported by the General Research Funds from Hong Kong Research Grants Council [Grants HKBU203613 and HKBU12300515.]

Keywords: convex programming • alternating direction method of multipliers • convergence rate • iteration complexity • contraction methods

1. Introduction

We consider a structured convex minimization model with linear constraints and a separable objective function:

$$\begin{aligned} &\min \sum_{i=1}^{m} \theta_i(x_i) \\ &\sum_{i=1}^{m} A_i x_i = b; \\ &x_i \in \mathcal{X}_i, \quad i = 1, \ldots, m, \end{aligned} \tag{1}$$

where $\theta_i: \Re^{n_i} \to \Re$ $(i = 1, \ldots, m)$ are convex functions and they are not necessarily smooth; $\mathcal{X}_i \subseteq \Re^{n_i}$ $(i = 1, \ldots, m)$ are closed-convex sets; $A_i \in \Re^{l \times n_i}$ $(i = 1, \ldots, m)$ are given matrices with full column ranks; and $b \in \Re^l$ is a given vector. Throughout, the solution set of (1) is assumed to be nonempty. Our discussion focuses on the particular case of (1) where $m \geq 3$, see, e.g., Peng et al. [35], Ruszczyński [39], Tao and Yuan [41], Tibshirani et al. [42], Zhou et al. [48] for such applications.

Except for problems in very small dimensions, it is not wise to treat (1) as a generic convex programming and ignore its favorable separable structure when we try to design efficient numerical algorithms for (1). One obvious reason is that each single θ_i could be simple, while the aggregation of all θ_i's is hard, to be minimized for many concrete applications of the abstract model (1). This has inspired many interesting results in the literature, e.g., a series of papers combining the augmented Lagrangian method (see Hestenes [23], Powell [36]), smoothing technique and Lagrangian duality theory in Lan and Monteiro [28], Necoara and Suykens [30], Nedelcu et al. [31], Tran-Dinh et al. [43, 44]. We are thus in favor of such an algorithm that can take advantage of the separable structure of (1) effectively; or more precisely, can exploit the properties of all θ_i's fully by treating these functions individually rather than aggregatively. In the literature, structure-exploited algorithms have been well studied, but only for the special case of (1) where $m = 2$. A fundamental method for this special case is the Douglas-Rachford

alternating direction method of multipliers (ADMM) proposed in Glowinski and Marrocco [16] (see also Gabay and Mercier [13]). More specifically, for solving (1) with $m = 2$, the standard ADMM iterative scheme is

$$\begin{cases} x_1^{k+1} = \arg\min\{\theta_1(x_1) - x_1^T A_1^T \lambda^k + \frac{1}{2}\|A_1 x_1 + A_2 x_2^k - b\|_H^2 \mid x_1 \in \mathscr{X}_1\}, \\ x_2^{k+1} = \arg\min\{\theta_2(x_2) - x_2^T A_2^T \lambda^k + \frac{1}{2}\|A_1 x_1^{k+1} + A_2 x_2 - b\|_H^2 \mid x_2 \in \mathscr{X}_2\}, \\ \lambda^{k+1} = \lambda^k - H(A_1 x_1^{k+1} + A_2 x_2^{k+1} - b), \end{cases} \tag{2}$$

where $\lambda^k \in \Re^l$ is the Lagrange multiplier and the penalty matrix $H \in \Re^{l\times l}$ is positive definite. In applications, we usually choose H as a diagonal matrix: $H = \beta I_{l\times l}$ with $\beta > 0$. The standard ADMM scheme (2) shows that ADMM blends the benefits of the dual decomposition and the augmented Lagrangian method. It is thus possible to exploit the properties of θ_i's individually. We refer to, e.g., Chan and Glowinski [4], Eckstein and Bertsekas [8], Fortin and Glowinski [11], Fukushima [12], Glowinski [14], Glowinski and Le Tallec [15], He et al. [22], Kontogiorgis and Meyer [27], Tseng [45] for some earlier articles in the areas of partial differential equations, convex programming and variational inequalities. Moreover, recently ADMM has found many interesting applications in a broad spectrum of areas such as imaging processing, statistical learning and engineering, see, e.g., Boyd et al. [2], Chan et al. [3], Chen et al. [5], Esser [9], He et al. [21], Ng et al. [33, 32], Sun and Zhang [40], Tao and Yuan [41], Zhang et al. [46] to mention a few. Essentially, the main reason ensuring ADMM's efficiency for these concrete applications (where the functions θ_i's usually have special properties) is that the decomposed subproblems in (2) are often simple enough to have closed-form solutions or can be solved efficiently up to high precisions. In the review paper on ADMM (Boyd et al. [2], p. 104), the authors complimented that "ADMM is at least comparable to very specialized algorithms (even in the serial setting), and in most cases, the simple ADMM algorithm will be efficient enough to be useful."

To take advantage of each θ_i's properties individually, a natural idea for solving the general case of (1) with $m \geq 3$ is to extend the ADMM scheme (2) straightforwardly—yielding the following scheme:

$$\begin{cases} x_1^{k+1} = \arg\min\left\{\theta_1(x_1) - x_1^T A_1^T \lambda^k + \frac{1}{2}\left\|A_1 x_1 + \sum_{j=2}^m A_j x_j^k - b\right\|_H^2 \,\middle|\, x_1 \in \mathscr{X}_1\right\}; \\ x_2^{k+1} = \arg\min\left\{\theta_2(x_2) - x_2^T A_2^T \lambda^k + \frac{1}{2}\left\|A_1 x_1^{k+1} + A_2 x_2 + \sum_{j=3}^m A_j x_j^k - b\right\|_H^2 \,\middle|\, x_2 \in \mathscr{X}_2\right\}; \\ \quad \cdots \\ x_i^{k+1} = \arg\min\left\{\theta_i(x_i) - x_i^T A_i^T \lambda^k + \frac{1}{2}\left\|\sum_{j=1}^{i-1} A_j x_j^{k+1} + A_i x_i + \sum_{j=i+1}^m A_j x_j^k - b\right\|_H^2 \,\middle|\, x_i \in \mathscr{X}_i\right\}; \\ \quad \cdots \\ x_m^{k+1} = \arg\min\left\{\theta_m(x_m) - x_m^T A_m^T \lambda^k + \frac{1}{2}\left\|\sum_{j=1}^{m-1} A_j x_j^{k+1} + A_m x_m - b\right\|_H^2 \,\middle|\, x_m \in \mathscr{X}_m\right\}; \\ \lambda^{k+1} = \lambda^k - H\left(\sum_{j=1}^m A_j x_j^{k+1} - b\right). \end{cases} \tag{3}$$

Just as the original ADMM scheme (2), the iterative scheme (3) can be easily derived by decomposing the augmented Lagrangian function of (1) in the Gauss-Seidel fashion. In (3), the variables x_i's are minimized in alternating order and the decomposed subproblems are much easier than the original problem (1) since only one function θ_i is involved in its x_i-subproblem. Then, the step of updating the Lagrange multiplier coordinates all these solutions to local small subproblems to find a solution to a global large problem. Note that the direct extension of the ADMM scheme (3) reduces to the augmented Lagrangian method in Hestenes [23], Powell [36], and the standard ADMM scheme (2) when $m = 1$ and $m = 2$ in (1), respectively.

The convergence of the scheme (3), however, had perplexed authors for a long time. On one hand, the scheme (3) empirically works well for some applications, see, e.g., Peng et al. [35], Tao and Yuan [41]. On the other hand, in the literature, its convergence could be shown only under some further assumptions. For example, in Han and Yuan [17], the convergence of (3) was shown under the conditions that all θ_i are strongly convex and the penalty parameter β (when $H = \beta I_{l\times l}$) should be chosen judiciously within a certain interval. Moreover, when each function θ_i in (1) is of particular structure and the update of λ^{k+1} in (3) is required to adopt a new stepsize rather than $H = \beta I_{l\times l}$, i.e.,

$$\lambda^{k+1} = \lambda^k - \tau\left(\sum_{j=1}^m A_j x_j^{k+1} - b\right), \tag{4}$$

where $\tau > 0$ is sufficiently small to fulfill certain error bound conditions, the resulting scheme was proved to be convergent in Hong and Luo [24]; some linear convergent results were also analyzed in Hong and Luo [24]. In fact, the scheme (3) but with the new update of λ in (4) can be regarded as an implementation of the dual-ascent method to the dual of (1) with a shrank stepsize. Most recently, a counterexample was given in Chen et al. [6] showing that scheme (3) is not necessarily convergent without further assumptions; and a sufficient condition ensuring the convergence of (3) was given therein.

In general, it is not easy to verify whether the stepsize τ in (4) is small enough to satisfy the desired error bound. We thus stick to the direct extension of ADMM (3) where the stepsize for updating λ is taken as the same as the penalty parameter (thus it is not necessarily very small) and the function θ_i's in (3) are only assumed to be generic nonsmooth convex functions, and study in which way the convergence of (3) can be derived. In He et al. [19], we have shown that the resulting sequence is convergent if the output of (3) is further corrected by a Gaussian back substitution procedure. The numerical efficiency of the Gaussian back substitution procedure, together with its superiority to some other relevant work based on (3), have been illustrated numerically in He et al. [19], Ng et al. [34].

The Gaussian back substitution procedure in He et al. [19] still requires to compute the inverses of $(A_i^T H A_i)$ for $i = 2, \ldots, m-1$ (see Equation (3.4) in He et al. [19]), which could be computationally expensive for generic A_i's arising in some image processing applications. Some elaboration will be given in Section 7.1. This paper is a further study on this theme, with the aim at proposing an algorithmic framework to combine an ADMM procedure with a substitution procedure. This substitution procedure can be in either a forward (i.e., correcting the output of the ADMM procedure in the order of $x_2^{k+1} \to x_3^{k+1} \to \cdots \to x_m^{k+1} \to \lambda^{k+1}$) or backward (i.e., correcting the output of the ADMM procedure in the order of $\lambda^{k+1} \to x_m^{k+1} \to x_{m-1}^{k+1} \to \cdots \to x_2^{k+1}$) fashion. Two concrete algorithms are thus derived from the algorithmic framework. We will show that these two algorithms both reduce to the original ADMM (2) for (1) with $m = 2$. The forward and backward substitution procedures require no computation of any matrix's inverse, and they are both much computationally cheaper than the Gaussian back substitution procedure in He et al. [19] (see Sections 5.1 and 5.2 for elaboration). Moreover, we will show that the convergence of this algorithmic framework can be easily proved from the contraction perspective, and its local linear convergence rate is provable if a certain error bound condition is assumed. Without such an error bound assumption, we can estimate their worst-case convergence rates measured by the iteration complexity for the new algorithms in the nonergodic sense.

The rest of the paper is organized as follows. In Section 2, we provide some preliminaries that are useful for further discussions; and summarize some notations, which will help us present our analysis simpler. In Section 3, we propose an ADMM procedure inspired by (3) and prove an inequality for the sequence generated by this ADMM procedure. In Section 4, we elaborate on the motivation of developing an appropriate substitution procedure in combination with the ADMM procedure proposed in Section 3. Then, in Section 5, we specify the general algorithmic framework in combination with an ADMM procedure and a substitution procedure into two concrete algorithms; prove their global convergence and linear convergence rates in a unified framework from the contraction perspective. In Section 6, we estimate the worst-case convergence rate measured by the iteration complexity for the proposed algorithms in the nonergodic sense. In Section 7, we report some numerical results when the new algorithms are applied to solve some applications of the model (1) arising in image processing. Finally, we make some conclusions in Section 8.

2. Preliminaries

In this section, we first provide some preliminaries, which are useful for our further discussions and then summarize some notations to be used.

2.1. A Variational Characterization of (1)

We first reformulate (1) as a variational form, which is useful for our subsequent analysis of convergence and the estimate of worst-case convergence rate measured by the iteration complexity.

Let $\lambda \in \Re^l$ be the Lagrange multiplier associated with the linear constraints in (1). Then, the Lagrangian function of (1) is

$$L(x_1, x_2, \ldots, x_m, \lambda) = \sum_{i=1}^{m} \theta_i(x_i) - \lambda^T \left(\sum_{i=1}^{m} A_i x_i - b \right),$$

and it is defined on

$$\mathcal{W} = \mathcal{X} \times \Re^l, \quad \text{where } \mathcal{X} = \mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_m.$$

Let $(x_1^*, x_2^*, \ldots, x_m^*, \lambda^*)$ be a saddle point of the Lagrangian function. Then, we have the saddle point inequalities

$$L_{\lambda \in \Re^l}(x_1^*, x_2^*, \ldots, x_m^*, \lambda) \le L(x_1^*, x_2^*, \ldots, x_m^*, \lambda^*) \le L_{x_i \in \mathcal{X}_i i=1,\ldots,m}(x_1, x_2, \ldots, x_m, \lambda^*). \tag{5}$$

For $i = 1, \ldots, m$, we set

$$(x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_m) = (x_1^*, \ldots, x_{i-1}^*, x_i, x_{i+1}^*, \ldots, x_m^*) \in \mathcal{X},$$

in the second inequality in (5). Then, we have

$$x_i^* \in \mathcal{X}_i, \quad \theta_i(x_i) - \theta_i(x_i^*) + (x_i - x_i^*)^T(-A_i^T\lambda^*) \ge 0, \quad \forall\, x_i \in \mathcal{X}_i, \quad i = 1, \ldots, m.$$

Moreover, it follows from the first inequality in (5) that

$$\lambda^* \in \Re^l, \qquad (\lambda - \lambda^*)^T\left(\sum_{i=1}^m A_i x_i^* - b\right) \ge 0, \quad \forall\, \lambda \in \Re^l.$$

Thus, finding a saddle point of $L(x_1, x_2, \ldots, x_m, \lambda)$ is equivalent to finding a vector

$$w^* = (x_1^*, x_2^*, \ldots, x_m^*, \lambda^*) \in \mathcal{W}$$

such that

$$\begin{cases} \theta_1(x_1) - \theta_1(x_1^*) + (x_1 - x_1^*)^T(-A_1^T\lambda^*) \ge 0, & \forall\, x_1 \in \mathcal{X}_1, \\ \theta_2(x_2) - \theta_2(x_2^*) + (x_2 - x_2^*)^T(-A_2^T\lambda^*) \ge 0, & \forall\, x_2 \in \mathcal{X}_2, \\ \quad \vdots & \\ \theta_m(x_m) - \theta_m(x_m^*) + (x_m - x_m^*)^T(-A_m^T\lambda^*) \ge 0, & \forall\, x_m \in \mathcal{X}_m, \\ (\lambda - \lambda^*)^T\left\{\sum_{i=1}^m A_i x_i^* - b\right\} \ge 0, & \forall\, \lambda \in \Re^l. \end{cases} \tag{6}$$

More compactly, the inequalities in (6) can be rewritten as the following variational inequality (VI):

$$\mathrm{VI}(\mathcal{W}, F, \theta)\text{: } w^* \in \mathcal{W}, \qquad \theta(x) - \theta(x^*) + (w - w^*)^T F(w^*) \ge 0, \quad \forall\, w \in \mathcal{W}, \tag{7a}$$

where

$$x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{pmatrix}, \qquad \theta(x) = \sum_{i=1}^m \theta_i(x_i), \qquad w = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \\ \lambda \end{pmatrix} \quad \text{and} \quad F(w) = \begin{pmatrix} -A_1^T\lambda \\ -A_2^T\lambda \\ \vdots \\ -A_m^T\lambda \\ \sum_{i=1}^m A_i x_i - b \end{pmatrix}. \tag{7b}$$

Note that the operator $F(w)$ defined in (7b) is monotone because it is affine with a skew-symmetric matrix. Let $\mathcal{W}^*$ be the solution set of $\mathrm{VI}(\mathcal{W}, F, \theta)$. Since the solution set of (1) is assumed to be nonempty, $\mathcal{W}^*$ is also nonempty.

2.2. Some Notations

In this subsection, we summarize some notations, which will be used in later analysis. These notations will make the presentation of our theoretical analysis in later sections more compact.

First, revisit the iterative scheme of the direct extension of ADMM (3). It is easy to notice that the iterate x_1^k is not involved in the $(k+1)$-th iteration of (3), just like the original ADMM (2). In other words, the input to execute the

iteration of (3) is only the sequence $\{x_2^k, \ldots, x_m^k, \lambda^k\}$. Therefore, following Boyd et al. [2], we call x_1 an intermediate variable. It is thus natural to introduce the notations

$$v = \begin{pmatrix} x_2 \\ \vdots \\ x_m \\ \lambda \end{pmatrix} \quad \text{and} \quad \mathcal{V} = \mathcal{X}_2 \times \cdots \times \mathcal{X}_m \times \Re^l$$

to differentiate the variables, which are truly involved in the iteration from the intermediate variable x_1. Obviously, v is a subvector of w defined in (7b). The notations v^k is thus clear from the context. Accordingly, we also use the notation

$$\mathcal{V}^* = \{(x_2^*, \ldots, x_m^*, \lambda^*) \mid (x_1^*, x_2^*, \ldots, x_m^*, \lambda^*) \in \mathcal{W}^*\}.$$

Taking a closer look at (3), we can easily find that the input to generate a new iterate is indeed $\{A_2x_2^k, \ldots, A_mx_m^k, \lambda^k\}$. Thus, it is convenient to define

$$u = \begin{pmatrix} u_2 \\ \vdots \\ u_m \\ u_\lambda \end{pmatrix} = \begin{pmatrix} H^{1/2}A_2x_2 \\ \vdots \\ H^{1/2}A_mx_m \\ H^{-1/2}\lambda \end{pmatrix} \tag{8}$$

from which the notation $u^k = [u_2^k, \ldots, u_m^k, u_\lambda^k]$ is also clear. For some concrete applications of (1) such as those in He et al. [19], Ng et al. [34], the matrices $A_2, \ldots A_m$ are all identity matrices. For these cases, u reduces to v if $H = I$.

Accordingly, we also use the notation

$$\mathcal{U}^* = \left\{ u^* = \begin{pmatrix} u_2^* \\ \vdots \\ u_m^* \\ u_\lambda^* \end{pmatrix} = \begin{pmatrix} H^{1/2}A_2x_2^* \\ \vdots \\ H^{1/2}A_mx_m^* \\ H^{-1/2}\lambda^* \end{pmatrix} \middle| \quad w^* = (x_1^*, \ldots, x_m^*, \lambda^*) \in \mathcal{W}^* \right\}.$$

With these notations, the scheme (3) can be summarized as generating the new iteration w^{k+1} with the input u^k.

Second, we introduce several matrices in blockwise form. With these matrices, our notation for theoretical analysis will be much easier and more compact. Recall that the variable x_1 is an intermediate variable for the scheme (3). We thus introduce the matrix

$$\mathcal{A} = \operatorname{diag}(H^{1/2}A_2, H^{1/2}A_3, \ldots, H^{1/2}A_m, H^{-1/2}I_l) \tag{9}$$

to collect all the coefficient matrices with respect to the variables in (1), except for x_1 and the identity matrix (corresponding to the Lagrange multiplier). With the definition (9), we obviously can relate the variables v and u as

$$u = \mathcal{A}v. \tag{10}$$

We also define the following block matrices:

$$\mathcal{S} = (I_l, I_l, \ldots, I_l, -I_l) \tag{11}$$

and

$$\mathcal{I} = \operatorname{diag}(I_l, I_l, \ldots, I_l, I_l). \tag{12}$$

In addition, we define the following $m \times m$-block matrices:

$$\mathcal{P} = \begin{pmatrix} I_l & 0 & \cdots & \cdots & 0 \\ I_l & I_l & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ I_l & \cdots & I_l & I_l & 0 \\ 0_l & \cdots & 0_l & 0_l & I_l \end{pmatrix}, \qquad \mathcal{N} = \begin{pmatrix} I_l & 0 & \cdots & \cdots & 0 \\ 0_l & I_l & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0_l & \cdots & 0_l & I_l & 0 \\ -I_l & \cdots & -I_l & -I_l & I_l \end{pmatrix}, \tag{13}$$

and

$$\mathscr{L}=\begin{pmatrix} I_l & 0 & \cdots & \cdots & 0 \\ I_l & I_l & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ I_l & \cdots & I_l & I_l & 0 \\ -I_l & \cdots & -I_l & -I_l & I_l \end{pmatrix}, \tag{14}$$

which will be used in the substitution procedures to be proposed. Note that $\mathscr{P}$, $\mathscr{N}$, and $\mathscr{L}$ are all well-structured block lower triangular matrices, and they are related in the following equation:

$$\mathscr{L}=\mathscr{P}\mathscr{N}. \tag{15}$$

Moreover, for (1) with $m=2$, we have $\mathscr{P}=\mathscr{I}$ and $\mathscr{N}=\mathscr{L}$.

Finally, two more matrices are useful:

$$\tilde{\mathscr{Q}}=\begin{pmatrix} 0 & 0 & \cdots & \cdots & 0 \\ A_2^TH^{1/2} & 0 & \cdots & \cdots & 0 \\ A_3^TH^{1/2} & A_3^TH^{1/2} & \ddots & & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ A_m^TH^{1/2} & A_m^TH^{1/2} & \cdots & A_m^TH^{1/2} & 0 \\ -H^{-1/2} & -H^{-1/2} & \cdots & -H^{-1/2} & H^{-1/2} \end{pmatrix} \tag{16}$$

and

$$\mathscr{Q}=\begin{pmatrix} A_2^TH^{1/2} & 0 & \cdots & \cdots & 0 \\ A_3^TH^{1/2} & A_3^TH^{1/2} & \ddots & & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ A_m^TH^{1/2} & A_m^TH^{1/2} & \cdots & A_m^TH^{1/2} & 0 \\ -H^{-1/2} & -H^{-1/2} & \cdots & -H^{-1/2} & H^{-1/2} \end{pmatrix}. \tag{17}$$

As we shall show, these two matrices are mainly used to represent the accumulated iterative information in a compact way and thus simplify our notation. In fact, $\tilde{\mathscr{Q}}$ is used as the associated matrix to left multiply a vector w^k, while $\mathscr{Q}$ is the associated matrix to left multiply a vector v^k. Note that the absence of the intermediate variable x_1^k in the iteration to be proposed also explains why the first row of $\tilde{\mathscr{Q}}$ is a zero vector. Nevertheless, despite that the matrix $\mathscr{Q}$ can be obtained by easily removing the first row of $\tilde{\mathscr{Q}}$, we give the explicit expressions of these two relevant matrices separately as we need to use both of them later.

Some relationships among the matrices defined above are summarized in the following lemmas. We omit their proofs since they are elementary.

Lemma 2.1. *Let the matrices $\mathscr{S}$, $\mathscr{I}$, and $\mathscr{L}$ be defined in* (11), (12), *and* (14), *respectively. Then, we have*

$$\mathscr{L}^T+\mathscr{L}=\mathscr{I}+\mathscr{S}^T\mathscr{S}. \tag{18}$$

Lemma 2.2. *Let the matrices $\mathscr{A}$, $\mathscr{L}$, and $\mathscr{Q}$ be defined in* (9), (14), *and* (17), *respectively. Then, we have*

$$\mathscr{Q}=\mathscr{A}^T\mathscr{L}. \tag{19}$$

The identities (18) and (19) in Lemmas 2.1 and 2.2 will be used in our theoretical analysis.

3. An ADMM Procedure Based On (3)

We have mentioned that an efficient strategy to overcome the divergence of the scheme (3) is to supplement a substitution procedure to the output of (3), as the algorithm in He et al. [19]. In this section, we propose an ADMM procedure, which is slightly different from the scheme (3) only in the update of its Lagrange multiplier.

3.1. An ADMM Procedure

Since the notations x_i^{k+1}'s will be used to denote the new iterate and the output of the proposing ADMM procedure is only its predictor, we use $\tilde{x}_i^k$'s to denote the output of the proposing ADMM procedure. Our ADMM procedure based on the scheme (3) is as follows, and it will be used as a subprocedure in the algorithms to be proposed.

Algorithm 1 (An ADMM Procedure (prediction step) for (1))

$$\begin{cases} \tilde{x}_1^k = \arg\min\left\{\theta_1(x_1) - x_1^T A_1^T \lambda^k + \frac{1}{2}\left\|A_1 x_1 + \sum_{j=2}^m A_j x_j^k - b\right\|_H^2 \,\middle|\, x_1 \in \mathcal{X}_1\right\}; \\ \tilde{x}_2^k = \arg\min\left\{\theta_2(x_2) - x_2^T A_2^T \lambda^k + \frac{1}{2}\left\|A_1 \tilde{x}_1^k + A_2 x_2 + \sum_{j=3}^m A_j x_j^k - b\right\|_H^2 \,\middle|\, x_2 \in \mathcal{X}_2\right\}; \\ \quad \cdots \\ \tilde{x}_i^k = \arg\min\left\{\theta_i(x_i) - x_i^T A_i^T \lambda^k + \frac{1}{2}\left\|\sum_{j=1}^{i-1} A_j \tilde{x}_j^k + A_i x_i + \sum_{j=i+1}^m A_j x_j^k - b\right\|_H^2 \,\middle|\, x_i \in \mathcal{X}_i\right\}; \\ \quad \cdots \\ \tilde{x}_m^k = \arg\min\left\{\theta_m(x_m) - x_m^T A_m^T \lambda^k + \frac{1}{2}\left\|\sum_{j=1}^{m-1} A_j \tilde{x}_j^k + A_m x_m - b\right\|_H^2 \,\middle|\, x_m \in \mathcal{X}_m\right\}. \end{cases} \tag{20a}$$

$$\tilde{\lambda}^k = \lambda^k - H\left(A_1 \tilde{x}_1^k + \sum_{j=2}^m A_j x_j^k - b\right). \tag{20b}$$

Remark 3.1. The scheme (20) differs from the direct extension of ADMM (3) (also the method in He et al. [19]) only in the way of updating the Lagrange multiplier, i.e., (20b), and all the essential subproblems dominating the computation (i.e., (20a)) at each iteration are the same as those in (3). For this reason, we still call the scheme (20) an ADMM procedure. The only difference in its update of the Lagrange multiplier indeed provides us the possibility to ensure the convergence for the combination of (20) with a desired substitution procedure. In fact, as we shall show in Theorem 3.3, this particular update (20b) enables us to derive an estimate on the accuracy of $\tilde{w}^k$ in (21), which is more succinct than the inequality (4.3) in He et al. [19]. To see the relationship between the outputs of (20) and (3), we set

$$x_1^{k+1} = \tilde{x}_1^k, \quad x_2^{k+1} = \tilde{x}_2^k, \ldots, x_m^{k+1} = \tilde{x}_m^k$$

and

$$\lambda^{k+1} = \tilde{\lambda}^k + H\left(\sum_{j=2}^m A_j (x_j^k - \tilde{x}_j^k)\right),$$

then the output of (3) is recovered. More compactly, by using the notations u, v, and $\mathcal{N}$, the outputs u^{k+1} generated by (3) and $\tilde{u}^k$ by (20) are related in the following way:

$$u^{k+1} = u^k - \mathcal{N}(u^k - \tilde{u}^k).$$

Remark 3.2. Taking a closer look at the iterative schemes (3) and (20), it is easy to find that, at each iteration, the input to implement these schemes is actually $(A_2 x_2^k, \ldots, A_m x_m^k, \lambda^k)$.

3.2. An Important Inequality

In the following theorem, we prove an important inequality for the output of the ADMM procedure (20), which will be used often in our further discussions, including the analysis of the substitution procedure in Section 5 and the analysis of the worst-case convergence rate in Section 6.

Theorem 3.3. *Let $\tilde{w}^k$ be generated by the ADMM procedure (20) with given v^k and $u = \mathcal{A}v$. Then, we have*

$$\tilde{w}^k \in \mathcal{W}, \quad \theta(x) - \theta(\tilde{x}^k) + (w - \tilde{w}^k)^T F(\tilde{w}^k) \geq (u - \tilde{u}^k)^T \mathcal{L}(u^k - \tilde{u}^k), \quad \forall\, w \in \mathcal{W}, \tag{21}$$

where $\mathcal{L}$ is defined in (14).

Proof. Since $\tilde{x}_i^k$ is the solution of (20a), for $i = 1, 2, \ldots, m$, we have

$$\tilde{x}_i^k \in \mathcal{X}_i, \quad \theta_i(x_i) - \theta_i(\tilde{x}_i^k) + (x_i - \tilde{x}_i^k)^T \left\{A_i^T \left[H\left(\sum_{j=1}^i A \tilde{x}_j^k + \sum_{j=i+1}^m A_j x_j^k - b\right) - \lambda^k\right]\right\} \geq 0, \quad \forall\, x_i \in \mathcal{X}_i.$$

Substituting $\tilde{\lambda}^k = \lambda^k - H(A_1 \tilde{x}_1^k + \sum_{j=2}^m A_j x_j^k - b)$ (see (20b)) into the above inequality, we obtain

$$\tilde{x}_i^k \in \mathcal{X}_i, \quad \theta_i(x_i) - \theta_i(\tilde{x}_i^k) + (x_i - \tilde{x}_i^k)^T \left\{-A_i^T \tilde{\lambda}^k + A_i^T H\left(\sum_{j=2}^i (A_j \tilde{x}_j^k - A_j x_j^k)\right)\right\} \geq 0, \quad \forall\, x_i \in \mathcal{X}_i.$$

Summarizing the above inequality over $i = 1, \ldots, m$, we obtain $\tilde{w}^k \in \mathcal{W}$ and

$$\theta(x) - \theta(\tilde{x}^k) + \begin{pmatrix} x_1 - \tilde{x}_1^k \\ x_2 - \tilde{x}_2^k \\ \cdots \\ x_i - \tilde{x}_i^k \\ \cdots \\ x_m - \tilde{x}_m^k \end{pmatrix}^T \left\{ \begin{pmatrix} -A_1^T\tilde{\lambda}^k \\ -A_2^T\tilde{\lambda}^k \\ \cdots \\ -A_i^T\tilde{\lambda}^k \\ \cdots \\ -A_m^T\tilde{\lambda}^k \end{pmatrix} + \begin{pmatrix} 0 \\ A_2^T H(A_2\tilde{x}_2^k - A_2x_2^k) \\ \cdots \\ A_i^T H\big(\sum_{j=2}^{i}(A_j\tilde{x}_j^k - A_jx_j^k)\big) \\ \cdots \\ A_m^T H\big(\sum_{j=2}^{m}(A_j\tilde{x}_j^k - A_jx_j^k)\big) \end{pmatrix} \right\} \geq 0, \quad \forall\, x_i \in \mathcal{X}_i. \tag{22}$$

Because $A\tilde{x}_1^k + \sum_{j=2}^m A_jx_j^k - b + H^{-1}(\tilde{\lambda}^k - \lambda^k) = 0$ (see (20b) again), we have

$$\left(\sum_{j=1}^{m} A_j\tilde{x}_j^k - b\right) + H^{-1}(\tilde{\lambda}^k - \lambda^k) - \sum_{j=2}^{m}(A_j\tilde{x}_j^k - A_jx_j^k) = 0. \tag{23}$$

Combining (22) and (23), we get $\tilde{w}^k \in \mathcal{W}$ and

$$\theta(x) - \theta(\tilde{x}^k) + \begin{pmatrix} x_1 - \tilde{x}_1^k \\ x_2 - \tilde{x}_2^k \\ \cdots \\ x_i - \tilde{x}_i^k \\ \cdots \\ x_m - \tilde{x}_m^k \\ \lambda^k - \tilde{\lambda}^k \end{pmatrix}^T \left\{ \begin{pmatrix} -A_1^T\tilde{\lambda}^k \\ -A_2^T\tilde{\lambda}^k \\ \cdots \\ -A_i^T\tilde{\lambda}^k \\ \cdots \\ -A_m^T\tilde{\lambda}^k \\ \sum_{j=1}^m A_j\tilde{x}_j^k - b \end{pmatrix} + \begin{pmatrix} 0 \\ A_2^T H(A_2\tilde{x}_2^k - A_2x_2^k) \\ \cdots \\ A_i^T H\big(\sum_{j=2}^{i}(A_j\tilde{x}_j^k - A_jx_j^k)\big) \\ \cdots \\ A_m^T H\big(\sum_{j=2}^{m}(A_j\tilde{x}_j^k - A_jx_j^k)\big) \\ H^{-1}(\tilde{\lambda}^k - \lambda^k) - \sum_{j=2}^{m}(A_j\tilde{x}_j^k - A_jx_j^k) \end{pmatrix} \right\} \geq 0$$

for all $w \in \mathcal{W}$. Using the notations of F (see (7b)), u (see (8)) and $\tilde{\mathcal{Q}}$ (see (16)), the above inequality can be rewritten as

$$\tilde{w}^k \in \mathcal{W}, \quad \theta(x) - \theta(\tilde{x}^k) + (w - \tilde{w}^k)^T\{F(\tilde{w}^k) - \tilde{\mathcal{Q}}(u^k - \tilde{u}^k)\} \geq 0, \quad \forall\, w \in \mathcal{W}. \tag{24}$$

Recall the definitions of $\tilde{\mathcal{Q}}$ and $\mathcal{Q}$. We then have

$$(w - \tilde{w}^k)^T\tilde{\mathcal{Q}}(u^k - \tilde{u}^k) = (v - \tilde{v}^k)^T\mathcal{Q}(u^k - \tilde{u}^k).$$

Because $\mathcal{Q} = \mathcal{A}^T\mathcal{L}$ and $\mathcal{A}v = u$ (see (19) and (10)), we have

$$(w - \tilde{w}^k)^T\tilde{\mathcal{Q}}(u^k - \tilde{u}^k) = (u - \tilde{u}^k)^T\mathcal{L}(u^k - \tilde{u}^k).$$

The assertion (21) thus follows from (24) and the last equality immediately. □

4. The Motivation of Finding a Substitution Procedure: From the Contraction Perspective

As we have mentioned, the output of (20) needs to be further corrected to yield convergence, and our strategy for the correction is to propose a substitution procedure whose computation is cheap. The idea inspiring the substitution strategy comes from the fact that the output of (20) can be corrected easily such that the corrected sequence is contractive with respect to the set $\mathcal{U}^*$. Note that we follow the classical definition of a contractive sequence in Blum and Oettli [1]. Then, we show that the correction step on contraction purpose can be easily executed in either a forward or backward substitution fashion, because of the particular structure of the matrix $\mathcal{Q}$. We would emphasize that Theorem 3.3 plays an important role for the coming analysis.

First, with the respective definitions of $\mathcal{A}$ and $\mathcal{L}$ in (9) and (14), we can prove two obvious propositions regarding the output of the ADMM procedure (20).

Proposition 4.1. *Let $\tilde{w}^k$ be generated by the ADMM procedure (20) with given v^k and $u = \mathcal{A}v$. Then, we have*

$$(u^k - u^*)^T\mathcal{L}(u^k - \tilde{u}^k) \geq (u^k - \tilde{u}^k)^T\mathcal{L}(u^k - \tilde{u}^k), \quad \forall\, u^* \in \mathcal{U}^*, \tag{25}$$

where $\mathcal{L}$ is defined in (14).

Proof. The proof is an immediate conclusion based on the assertion (21) and the monotonicity of F. In fact, for an arbitrarily fixed $w^* \in \mathcal{W}^*$, it follows from (21) that

$$(\tilde{u}^k - u^*)^T \mathcal{L}(u^k - \tilde{u}^k) \geq (\tilde{w}^k - w^*)^T F(\tilde{w}^k) + \theta(\tilde{x}^k) - \theta(x^*), \quad \forall\, w^* \in \mathcal{W}^*.$$

Using the monotonicity of F and the optimality of w^*, we have

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

The above two inequalities imply that

$$(\tilde{u}^k - u^*)^T \mathcal{L}(u^k - \tilde{u}^k) \geq 0, \quad \forall\, u^* \in \mathcal{U}^*,$$

and the assertion (25) follows from the last inequality immediately. □

Proposition 4.2. *Let $\tilde{w}^k$ be generated by the ADMM procedure (20) with given v^k and $u = \mathcal{A}v$. Then, we have*

$$(u^k - \tilde{u}^k)^T \mathcal{L}(u^k - \tilde{u}^k) = \tfrac{1}{2}\|u^k - \tilde{u}^k\|^2 + \tfrac{1}{2}\|\mathcal{S}(u^k - \tilde{u}^k)\|^2, \tag{26}$$

where $\mathcal{L}$ and $\mathcal{S}$ are defined in (14) and (11), respectively.

Proof. First, we have

$$(u^k - \tilde{u}^k)^T \mathcal{L}(u^k - \tilde{u}^k) = \tfrac{1}{2}(u^k - \tilde{u}^k)^T (\mathcal{L}^T + \mathcal{L})(u^k - \tilde{u}^k).$$

Then, using the identity (18) in Lemma 2.1, the assertion (26) is proved immediately. □

Remark 4.3. The assertions (21) and (26) jointly imply that if $\|u^k - \tilde{u}^k\| = 0$, then we have

$$\tilde{w}^k \in \mathcal{W}, \quad \theta(x) - \theta(\tilde{x}^k) + (w - \tilde{w}^k)^T F(\tilde{w}^k) \geq 0, \quad \forall\, w \in \mathcal{W},$$

which means that the output $\tilde{w}^k$ of the ADMM procedure (20) is a solution point of VI($\mathcal{W}, F, \theta$) (see (7)). Therefore the result (26) provides us an easy stopping criterion to terminate the iteration (20):

$$\|u^k - \tilde{u}^k\| \leq \epsilon,$$

where ϵ is the tolerance set by users.

Now, with Propositions 4.1 and 4.2, we have

$$(u^k - u^*)^T \mathcal{L}(u^k - \tilde{u}^k) \geq \tfrac{1}{2}\|u^k - \tilde{u}^k\|^2 + \tfrac{1}{2}\|\mathcal{S}(u^k - \tilde{u}^k)\|^2, \quad \forall\, u^* \in \mathcal{U}^*, \tag{27}$$

and it becomes apparent how to correct the output of (20) in the contraction way. More specifically, whenever $\|u^k - \tilde{u}^k\| \neq 0$, the assertion (26) shows the positivity of the term $(u^k - \tilde{u}^k)^T \mathcal{L}(u^k - \tilde{u}^k)$, and thus the assertion (25) indicates that the direction $-\mathcal{L}(u^k - \tilde{u}^k)$ is beneficial for reducing the proximity to the solution set $\mathcal{U}^*$ if the current iterate $\tilde{u}^k$ moves along this direction with an appropriate stepsize. That is, the spirit of contraction-type methods (see Blum and Oettli [1]) is applicable. More explicitly, the new iterate u^{k+1} can be generated by

$$u^{k+1} = u^k - \alpha G^{-1} \mathcal{L}(u^k - \tilde{u}^k), \tag{28}$$

where G is an arbitrarily symmetric positive definite matrix with the same dimensionality as $\mathcal{L}$ and $\alpha > 0$ is an undetermined stepsize. Then, with an appropriate choice of the stepsize α, we can prove that the sequence $\{u^k\}$ generated by (28) is contractive with respect to the set $\mathcal{U}^*$ under the G-norm. In other words, the scheme (28) can be used to correct the output of (20).

To see why the scheme (28) yields a contractive sequence, we have the following easy fact:

$$\begin{aligned} \|u^k - u\|_G^2 - \|u^{k+1} - u\|_G^2 &= \|u^k - u\|_G^2 - \|(u^k - u) - \alpha G^{-1}\mathcal{L}(u^k - \tilde{u}^k)\|_G^2 \\ &= 2\alpha(u^k - u)^T \mathcal{L}(u^k - \tilde{u}^k) - \alpha^2 \|G^{-1}\mathcal{L}(u^k - \tilde{u}^k)\|_G^2. \end{aligned} \tag{29}$$

Set $u = u^*$ in (29), and use (25) and (26). Then, we get

$$\begin{aligned} \|u^k - u^*\|_G^2 - \|u^{k+1} - u^*\|_G^2 &\geq 2\alpha(u^k - \tilde{u}^k)^T \mathcal{L}(u^k - \tilde{u}^k) - \alpha^2 \|G^{-1}\mathcal{L}(u^k - \tilde{u}^k)\|_G^2 \\ &= \alpha(\|u^k - \tilde{u}^k\|^2 + \|\mathcal{S}(u^k - \tilde{u}^k)\|^2) - \alpha^2 \|G^{-1}\mathcal{L}(u^k - \tilde{u}^k)\|_G^2, \quad \forall\, u^* \in \mathcal{U}^*. \end{aligned} \tag{30}$$

Note that right-hand side of (30) is a quadratic function of α. To obtain the closest proximity to $\mathcal{U}^*$, we wish to maximize this quadratic function, and this desire inspires us to take the optimal value of α as

$$\alpha = \alpha_k := \frac{\|u^k - \tilde{u}^k\|^2 + \|\mathcal{S}(u^k - \tilde{u}^k)\|^2}{2\|G^{-1}\mathcal{L}(u^k - \tilde{u}^k)\|_G^2}. \tag{31}$$

With this choice of stepsize, it follows from (30) that

$$\|u^{k+1} - u^*\|_G^2 \le \|u^k - u^*\|_G^2 - \tfrac{1}{2}\alpha_k(\|u^k - \tilde{u}^k\|^2 + \|\mathcal{S}(u^k - \tilde{u}^k)\|^2), \quad \forall\, u^* \in \mathcal{U}^*. \tag{32}$$

Since $\alpha_k > 0$ whenever $\|u^k - \tilde{u}^k\| \neq 0$, (32) shows that the sequence $\{u^k\}$ generated by the scheme (28) is contractive with respect to $\mathcal{U}^*$ if the stepsize is appropriately chosen.

Indeed, it is flexible to choose different positive definite matrices for G in the generic scheme (28) if our purpose is only to ensure that the sequence $\{u^k\}$ is contractive with respect to $\mathcal{U}^*$. To induce simple substitution procedures, which are computationally inexpensive, two specific interesting choices are

$$G = \mathcal{I} \quad \text{and} \quad G = \mathcal{P}\mathcal{P}^T.$$

In fact, these two choices yield two efficient substitution procedures to correct the output of (20), as we elaborate below.

- If $G = \mathcal{I}$, then the scheme (28) reduces to

$$u^{k+1} - u^k = \alpha\mathcal{L}(\tilde{u}^k - u^k), \tag{33}$$

and it can be rewritten as

$$\mathcal{L}^{-1}(u^{k+1} - u^k) = \alpha(\tilde{u}^k - u^k). \tag{34}$$

Recall that the matrix $\mathcal{L}$ defined in (14) is a block unit lower triangular matrix and

$$\mathcal{L}^{-1} = \begin{pmatrix} I_l & 0 & \cdots & \cdots & 0 \\ -I_l & I_l & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & -I_l & I_l & 0 \\ 0 & \cdots & 0 & I_l & I_l \end{pmatrix}.$$

Thus the implementation of (34) is essentially a forward substitution procedure where the new iterate u^{k+1} is yielded in the order of $u_2^{k+1} \to u_3^{k+1} \to \cdots \to u_m^{k+1} \to \lambda^{k+1}$. Overall, with the given u^k and the output $\tilde{u}^k$ of (20), the new iterative u^{k+1} can be generated via the forward substitution procedure (33). The resulting sequence $\{u^k\}$ is contractive with respect to the set $\mathcal{U}^*$ provided that the stepsize α is chosen appropriately.

- If $G = \mathcal{P}\mathcal{P}^T$ in (28), because the inequality (27) can be written as

$$(G(u^k - u^*))^T(G^{-1}\mathcal{L}(u^k - \tilde{u}^k)) \ge \tfrac{1}{2}\|u^k - \tilde{u}^k\|^2 + \tfrac{1}{2}\|\mathcal{S}(u^k - \tilde{u}^k)\|^2, \quad \forall\, u^* \in \mathcal{U}^*, \tag{35}$$

we can view $G^{-1}\mathcal{L}(\tilde{u}^k - u^k)$ as a descent direction of $\|u - u^*\|_G^2$ at the point u^k. By using $G = \mathcal{P}\mathcal{P}^T$ and (15), the scheme (28) can be rewritten as

$$\mathcal{P}^T(u^{k+1} - u^k) = \alpha\mathcal{P}^{-1}\mathcal{L}(\tilde{u}^k - u^k) = \alpha\mathcal{N}(\tilde{u}^k - u^k). \tag{36}$$

Recall the definition of $\mathcal{P}$ in (13). We see that (36) is essentially a backward substitution procedure to yield the new iterate u^{k+1} in the order of $\lambda^{k+1} \to u_m^{k+1} \to u_{m-1}^{k+1} \to \cdots \to u_2^{k+1}$. Also, the resulting sequence $\{u^k\}$ is contractive with respect to the set $\mathcal{U}^*$, provided that the stepsize α is chosen appropriately.

Therefore, a unified algorithmic framework combining the ADMM procedure (20) with the substitution procedure (28) is proposed for solving (1). In particular, we can choose the specific substitution procedures (33) and (36).

5. ADMM With a Substitution Procedure

As we have analyzed, the resulting algorithms, by combining the ADMM procedure (20) with the substitution procedure (28), are contraction-type methods if the involving stepsizes are chosen appropriately; thus their convergence analysis can be conducted from the contraction perspective. In this section, we specify the choices of stepsize in (33) and (36), and thus derive two concrete algorithms for solving (1). Then, we prove their global convergence in an unified way, by following the standard analytic framework of contraction methods.

5.1. ADMM With a Forward Substitution

We first combine the ADMM procedure (20) with the forward substitution (33) for solving (1). The resulting algorithm can be summarized as follows.

Algorithm 2 (ADMM with a forward substitution procedure)
Let $\gamma \in (0,2)$, $\mathscr{A}$, $\mathscr{S}$, and $\mathscr{L}$ be defined in (9), (11), and (14), respectively. Start with u^0. With the given iterate u^k, the new iterate u^{k+1} is given as follows.
Step 1. ADMM procedure (prediction step). Execute the scheme (20) to generate $\tilde{w}^k$ and thus $\tilde{v}^k$. Set $\tilde{u}^k = \mathscr{A}\tilde{v}^k$.
Step 2. Forward substitution procedure (correction step). Generate the new iterate u^{k+1} via.

$$u^{k+1} - u^k = \alpha_k \mathscr{L}(\tilde{u}^k - u^k), \tag{37a}$$

where

$$\alpha_k = \gamma \alpha_k^F \quad \text{with } \alpha_k^F = \frac{\|u^k - \tilde{u}^k\|^2 + \|\mathscr{S}(u^k - \tilde{u}^k)\|^2}{2\|\mathscr{L}(u^k - \tilde{u}^k)\|^2}. \tag{37b}$$

Remark 5.1. The strategy of determining α_k^F in (37b) is obtained by taking $G = \mathscr{I}$ in (31). As we have shown, it is for the purpose of ensuring that the sequence $\{u^k\}$ is contractive with respect to the set $\mathscr{U}^*$. Recall we determine the stepsize α by maximizing the quadratic function of α in the right-hand side of (30), which is merely a lower bound of the true proximity progress. Thus the optimal value of α calculated based on this quadratic function should be relaxed to some extent, and this is the role of the relaxation factor γ. For restricting γ into the interval $(0,2)$, it is to ensure the contractive property of the iterative sequence $\{u^k\}$, see (42).

Recall that the proposed Algorithm 2 is gained by taking $G = \mathscr{I}$ in (28). Thus the contraction of the sequence $\{u^k\}$ generated by Algorithm 2 is an immediate conclusion of the inequalities (30) and (32). For completeness, we summarize these results in the following theorem for the special case of $G = \mathscr{I}$. First, we emphasize that the forward substitution procedure with an undetermined stepsize $\alpha > 0$ is

$$u^{k+1} = u^k + \alpha \mathscr{L}(\tilde{u}^k - u^k). \tag{38}$$

Theorem 5.2. *Let $\tilde{w}^k$ be generated by the ADMM procedure (20) with given u^k and $\tilde{u}^k = \mathscr{A}\tilde{v}^k$. If the new iterate u^{k+1} is updated by (38), then we have*

$$\|u^k - u^*\|^2 - \|u^{k+1} - u^*\|^2 \geq q^F(\alpha), \quad \forall\, u^* \in \mathscr{U}^*, \tag{39}$$

where

$$q^F(\alpha) = \alpha(\|u^k - \tilde{u}^k\|^2 + \|\mathscr{S}(u^k - \tilde{u}^k)\|^2) - \alpha^2 \|\mathscr{L}(u^k - \tilde{u}^k)\|^2. \tag{40}$$

Proof. This is a special case of (30) by taking $G = \mathscr{I}$. □

Now, we are ready to show that the sequence $\{u^k\}$ generated by the proposed Algorithm 2 is contractive with respect to the set $\mathscr{U}^*$, starting from the following lemma.

Lemma 5.3. *Let α_k^F be defined in (37b). Then, we have*

$$\alpha_k^F \geq \frac{1}{2\|\mathscr{L}^T\mathscr{L}\|}. \tag{41}$$

Proof. It is trivial from (37b). □

Theorem 5.4. *Let the sequence $\{u^k\}$ be generated by the proposed Algorithm 2. Then, we have*

$$\|u^{k+1} - u^*\|^2 \leq \|u^k - u^*\|^2 - \frac{\gamma(2-\gamma)}{4\|\mathscr{L}^T\mathscr{L}\|}\|u^k - \tilde{u}^k\|^2, \quad \forall\, u^* \in \mathscr{U}^*. \tag{42}$$

Proof. Using (40), by a manipulation, we obtain

$$\begin{aligned} q^F(\alpha_k) &= \gamma\alpha_k^F(\|u^k - \tilde{u}^k\|^2 + \|\mathscr{S}(u^k - \tilde{u}^k)\|^2) - \gamma^2\alpha_k^F(\alpha_k^F\|\mathscr{L}(u^k - \tilde{u}^k)\|^2) \\ &\overset{(37b)}{=} \gamma\left(1 - \frac{\gamma}{2}\right)\alpha_k^F(\|u^k - \tilde{u}^k\|^2 + \|\mathscr{S}(u^k - \tilde{u}^k)\|^2) \\ &\overset{(41)}{\geq} \frac{\gamma(2-\gamma)}{4\|\mathscr{L}^T\mathscr{L}\|}\|u^k - \tilde{u}^k\|^2. \end{aligned}$$

The assertion (42) follows from (39) immediately. □

Remark 5.5. For the special case where $m = 2$, if we determine the stepsize α_k in a simpler way: $\alpha_k \equiv 1$, then the forward substitution procedure (37a) becomes

$$u^{k+1} - u^k = \mathscr{L}(\tilde{u}^k - u^k);$$

or more specifically,

$$\begin{pmatrix} H^{1/2}A_2(x_2^{k+1} - x_2^k) \\ H^{-1/2}(\lambda^{k+1} - \lambda^k) \end{pmatrix} = \begin{pmatrix} I_l & 0 \\ -I_l & I_l \end{pmatrix} \begin{pmatrix} H^{1/2}A_2(\tilde{x}_2^k - x_2^k) \\ H^{-1/2}(\tilde{\lambda}^k - \lambda^k) \end{pmatrix}. \tag{43}$$

In this case, we have (see (40))

$$\begin{aligned} q^F(1) &= \|u^k - \tilde{u}^k\|^2 + \|\mathscr{S}(u^k - \tilde{u}^k)\|^2 - \|\mathscr{L}(u^k - \tilde{u}^k)\|^2 \\ &= \left\| \begin{matrix} H^{1/2}A_2(x_2^k - \tilde{x}_2^k) \\ H^{-1/2}(\lambda^k - \tilde{\lambda}^k) \end{matrix} \right\|^2 + \|H^{1/2}A_2(x_2^k - \tilde{x}_2^k) - H^{-1/2}(\lambda^k - \tilde{\lambda}^k)\|^2 - \left\| \begin{pmatrix} I_l & 0 \\ -I_l & I_l \end{pmatrix} \begin{pmatrix} H^{1/2}A_2(x_2^k - \tilde{x}_2^k) \\ H^{-1/2}(\lambda^k - \tilde{\lambda}^k) \end{pmatrix} \right\|^2 \\ &= \|\lambda^k - \tilde{\lambda}^k\|_{H^{-1}}^2, \end{aligned} \tag{44}$$

and thus $q^F(1) \geq 0$. Combining (39) with the last equality, we obtain

$$\left\| \begin{matrix} H^{1/2}A_2(x_2^{k+1} - x_2^*) \\ H^{-1/2}(\lambda^{k+1} - \lambda^*) \end{matrix} \right\|^2 \leq \left\| \begin{matrix} H^{1/2}A_2(x_2^k - x_2^*) \\ H^{-1/2}(\lambda^k - \lambda^*) \end{matrix} \right\|^2 - \|\lambda^k - \tilde{\lambda}^k\|_{H^{-1}}^2, \quad \forall\, u^* \in \mathscr{U}^*.$$

Because

$$u^0 = \begin{pmatrix} H^{1/2}A_2x_2^0 \\ H^{-1/2}\lambda^0 \end{pmatrix}, \quad \tilde{u}^k = \begin{pmatrix} H^{1/2}A_2\tilde{x}_2^k \\ H^{-1/2}\tilde{\lambda}^k \end{pmatrix}, \quad \text{and} \quad A_2x_2^{k+1} = A_2\tilde{x}_2^k \qquad \text{(see (43))},$$

combining the definition of $\tilde{\lambda}^k$ (see (20b)), the method is equivalent to

$$\begin{cases} x^{k+1} = \tilde{x}^k, \\ \lambda^{k+1} = \lambda^k - H(A_1x_1^{k+1} + A_2x_2^{k+1} - b), \end{cases}$$

which is exactly the ADMM scheme (2). Therefore, for the case $m = 2$, Algorithm 2 with $\alpha_k \equiv 1$ is exactly the standard ADMM. Based on the above analysis, the proposed Algorithm 2 can be regarded as an extension of the standard ADMM scheme (2) from the special case with $m = 2$ to the general case with a generic m in (1).

Note that the computation of the stepsize α_k^F in (37b) is cheap, because $\mathscr{S}$ and $\mathscr{L}$ are very simple (see (11) and (14)). Moreover, in (41), we have shown a unified lower bound of this stepsize for any m. This bound is certainly conservative because it holds for the general case with a generic m. For a specific m, we are interested in a bound that is more accurate than this conservative bound. Also, one may ask the question if the stepsize α_k can be fixed as a constant, so that the computation of the forward substitution procedure (37a) can be further alleviated. To answer these questions, we need to analyze how to ensure the positivity of the term $q^F(\alpha)$. In the remark above, we has shown that α_k could be fixed as 1 for the special case where $m = 2$ in (1).

Moreover, according to (40), we have

$$q^F(\alpha) \geq 0 \iff \|u^k - \tilde{u}^k\|^2 + \|\mathscr{S}(u^k - \tilde{u}^k)\|^2 \geq \alpha\|\mathscr{L}(u^k - \tilde{u}^k)\|^2.$$

Hence we need only to take α such that

$$(\mathscr{I} + \mathscr{S}^T\mathscr{S}) - \alpha\mathscr{L}^T\mathscr{L} \geq 0$$

to ensure $q^F(\alpha) \geq 0$ (thus to ensure the contraction and convergence). For any fixed $m \geq 3$, we define

$$\alpha_*^F = \sup\{\alpha \mid (\mathscr{I} + \mathscr{S}^T\mathscr{S}) - \alpha\mathscr{L}^T\mathscr{L} \geq 0\}. \tag{45}$$

We summarize how to choose a constant stepsize for Algorithm 2 in the following theorem.

Theorem 5.6. *Let $\{u^k\}$ be the sequence generated by the proposed Algorithm 2 with a constant stepsize $\alpha \in (0, \alpha_*^F)$, where α_*^F is defined in (45). Then, we have*

$$q^F(\alpha) \geq \alpha(\alpha_*^F - \alpha)\|\mathscr{L}(u^k - \tilde{u}^k)\|^2, \tag{46}$$

and consequently,

$$\|u^{k+1} - u^*\|^2 \leq \|u^k - u^*\|^2 - \alpha(\alpha_*^F - \alpha)\|\mathscr{L}(u^k - \tilde{u}^k)\|^2, \quad \forall\, u^* \in \mathscr{U}^*. \tag{47}$$

Proof. Using (40), we have

$$q^F(\alpha) = \alpha((\|u^k - \tilde{u}^k\|^2 + \|\mathcal{S}(u^k - \tilde{u}^k)\|^2) - \alpha\|\mathcal{L}(u^k - \tilde{u}^k)\|^2).$$

According to (45), we have

$$(\|u^k - \tilde{u}^k\|^2 + \|\mathcal{S}(u^k - \tilde{u}^k)\|^2) \geq \alpha_*^F \|\mathcal{L}(u^k - \tilde{u}^k)\|^2$$

and thus (46) is proved. The assertion (47) follows from (39) and (46) immediately. □

Now, we take the case where $m = 3$ as an illustrative example. For this case, since

$$\mathcal{I} + \mathcal{S}^T\mathcal{S} = \begin{pmatrix} 2I & I & -I \\ I & 2I & -I \\ -I & -I & 2I \end{pmatrix} \quad \text{and} \quad \mathcal{L}^T\mathcal{L} = \begin{pmatrix} 3I & 2I & -I \\ 2I & 2I & -I \\ -I & -I & I \end{pmatrix},$$

we investigate the determinate of the matrix $(\mathcal{I} + \mathcal{S}^T\mathcal{S}) - \alpha\mathcal{L}^T\mathcal{L}$,

$$\Delta(\alpha) := \begin{vmatrix} 2-3\alpha & 1-2\alpha & \alpha-1 \\ 1-2\alpha & 2-2\alpha & \alpha-1 \\ \alpha-1 & \alpha-1 & 2-\alpha \end{vmatrix}. \tag{48}$$

In fact, with a detailed manipulation, we have

$$\begin{aligned} \Delta(\alpha) &= -2(\alpha-1)(\alpha-2)(3\alpha-2) + \{(2(1-2\alpha) + (2\alpha-2) + (3\alpha-2))\}(\alpha-1)^2 + (\alpha-2)(2\alpha-1)^2 \\ &= -2(\alpha-1)(\alpha-2)(3\alpha-2) + (\alpha-2)(\alpha-1)^2 + (\alpha-2)(2\alpha-1)^2 \\ &= -(\alpha-2)\{2(\alpha-1)(3\alpha-2) - (\alpha-1)^2 - (2\alpha-1)^2\} \\ &= -(\alpha-2)(\alpha^2 - 4\alpha + 2) \\ &= -(\alpha-(2-\sqrt{2}))(\alpha-2)(\alpha-(2+\sqrt{2})). \end{aligned}$$

Therefore we obtain

$$\Delta(\alpha) \geq 0, \quad \forall\, \alpha \in (0, 2-\sqrt{2}).$$

In addition, for such $\alpha \in (0, 2-\sqrt{2})$, it is easy to check that

$$2 - 3\alpha > 0 \qquad \text{and} \qquad \begin{vmatrix} 2-3\alpha & 1-2\alpha \\ 1-2\alpha & 2-2\alpha \end{vmatrix} > 0.$$

Thus

$$(\mathcal{I} + \mathcal{S}^T\mathcal{S}) - \alpha\mathcal{L}^T\mathcal{L} \geq 0, \quad \forall\, \alpha \in (0, (2-\sqrt{2})].$$

For $m = 3$, $\alpha_*^F = 2 - \sqrt{2}$. Therefore, to use the forward substitution in (37a), we can take fixed $\alpha_k \equiv \alpha \in (0, 2-\sqrt{2})$ if $m = 3$ in (1).

Although it saves computation by taking a constant stepsize, we have found some applications where the stepsize chosen by (37b) can accelerate the convergence of Algorithm 2, see the numerical results in Section 7.1. To implement Algorithm 2, our recommendation is to use a constant stepsize if the computation of (37b) is computationally expensive; or just use (37b) otherwise.

5.2. ADMM with a Backward Substitution

In this subsection, we combine the ADMM procedure (20) with the backward substitution (36) for solving (1). The resulting algorithm can be summarized as follows.

Algorithm 3 (ADMM with a Backward Substitution)
Let $\gamma \in (0, 2)$, $\mathcal{A}$, $\mathcal{S}$, and $\mathcal{L}$ be defined in (9), (11), and (14), respectively. Start with u^0. With the given iterate u^k, the new iterate u^{k+1} is given as follows.
Step 1. ADMM procedure (prediction step). Execute the scheme (20) to generate $\tilde{w}^k$ and thus $\tilde{v}^k$. Set $\tilde{u}^k = \mathcal{A}\tilde{v}^k$.
Step 2. Backward substitution procedure (correction step). Generate the new iterate u^{k+1} via

$$\mathcal{P}^T(u^{k+1} - u^k) = \alpha_k \mathcal{N}(\tilde{u}^k - u^k), \tag{49a}$$

where

$$\alpha_k = \gamma\alpha_k^B \quad \text{with } \alpha_k^B = \frac{\|u^k - \tilde{u}^k\|^2 + \|\mathcal{S}(u^k - \tilde{u}^k)\|^2}{2\|\mathcal{N}(u^k - \tilde{u}^k)\|^2}. \tag{49b}$$

To show how to choose the stepsize in the backward substitution procedure (49a), we first look at a backward substitution procedure whose stepsize α is undetermined, and then investigate how to seek an appropriate value for α on contraction purpose.

Theorem 5.7. *Let $\tilde{w}^k$ be generated by the ADMM procedure (20) with given u^k and $\tilde{u}^k = \mathscr{A}\tilde{v}^k$. If the new iterate u^{k+1} is updated by (49a), then we have*

$$\|u^k - u^*\|_G^2 - \|u^{k+1} - u^*\|_G^2 \geq q^B(\alpha), \quad \forall\, u^* \in \mathcal{U}^*, \tag{50}$$

where $G = \mathscr{P}\mathscr{P}^T$, $\mathscr{P}$ is defined in (13) and

$$q^B(\alpha) = \alpha(\|u^k - \tilde{u}^k\|^2 + \|\mathscr{S}(u^k - \tilde{u}^k)\|^2) - \alpha^2\|\mathscr{N}(u^k - \tilde{u}^k)\|^2. \tag{51}$$

Proof. Recall that (49a) is a special case of (28) by taking $G = \mathscr{P}\mathscr{P}^T$. It follows from (30) that

$$\|u^k - u^*\|_G^2 - \|u^{k+1} - u^*\|_G^2 \geq \alpha(\|u^k - \tilde{u}^k\|^2 + \|\mathscr{S}(u^k - \tilde{u}^k)\|^2) - \alpha^2\|\mathscr{P}^{-1}\mathscr{L}(u^k - \tilde{u}^k)\|^2, \quad \forall\, u^* \in \mathcal{U}^*. \tag{52}$$

Since $\mathscr{P}^{-1}\mathscr{L} = \mathscr{N}$ (see (15)), (50)–(51) is an immediate conclusion of (52). □

Accordingly, the inequality (50) suggests us to choose a value of α such that $q^B(\alpha)$ defined in (51) is maximized, i.e.,

$$\alpha_k^B = \frac{\|u^k - \tilde{u}^k\|^2 + \|\mathscr{S}(u^k - \tilde{u}^k)\|^2}{2\|\mathscr{N}(u^k - \tilde{u}^k)\|^2}.$$

Note that we can also take $G = \mathscr{P}\mathscr{P}^T$ in (31) and obtain the choice of α_k^B as above. Similar as the forward substitution procedure (37a), we attach a relaxation factor $\gamma \in (0, 2)$ to α_k^B, and thus choose the stepsize as

$$\alpha_k = \gamma\alpha_k^B$$

for the backward substitution procedure (49a).

Proposition 5.8. *For the $q^B(\alpha)$ defined in (51), we have*

$$q^B(\alpha) \geq \alpha\|\lambda^k - \tilde{\lambda}^k\|_{H^{-1}}^2, \quad \forall\, \alpha \in (0, 1] \tag{53}$$

and

$$q^B(\alpha) \geq \alpha(1 - \alpha)\|\mathscr{N}(u^k - \tilde{u}^k)\|^2, \quad \forall\, \alpha \in (0, 1]. \tag{54}$$

Proof. Using the structures of $\mathscr{N}$, u, and $\mathscr{S}$ (see (13), (8), and (11)), we have

$$\|\mathscr{N}(u^k - \tilde{u}^k)\|^2 = \|u^k - \tilde{u}^k\|^2 - \|\lambda^k - \tilde{\lambda}^k\|_{H^{-1}}^2 + \|\mathscr{S}(u^k - \tilde{u}^k)\|^2. \tag{55}$$

According to the definition of $q^B(\alpha)$, for any $\alpha \in (0, 1]$, we have

$$\begin{aligned} q^B(\alpha) &= \alpha\{(\|u^k - \tilde{u}^k\|^2 + \|\mathscr{S}(u^k - \tilde{u}^k)\|^2) - \alpha\|\mathscr{N}(u^k - \tilde{u}^k)\|^2\} \\ \text{(using (55))} &\geq \alpha\{(\|\mathscr{N}(u^k - \tilde{u}^k)\|^2 + \|\lambda^k - \tilde{\lambda}^k\|_{H^{-1}}^2) - \|\mathscr{N}(u^k - \tilde{u}^k)\|^2\} \\ &= \alpha\|\lambda^k - \tilde{\lambda}^k\|_{H^{-1}}^2. \end{aligned}$$

Using (see (55))

$$\|u^k - \tilde{u}^k\|^2 + \|\mathscr{S}(u^k - \tilde{u}^k)\|^2 \geq \|\mathscr{N}(u^k - \tilde{u}^k)\|^2 \tag{56}$$

for any $\alpha \in (0, 1]$, we have

$$\begin{aligned} q^B(\alpha) &= \alpha\{(\|u^k - \tilde{u}^k\|^2 + \|\mathscr{S}(u^k - \tilde{u}^k)\|^2) - \alpha\|\mathscr{N}(u^k - \tilde{u}^k)\|^2\} \\ &\geq \alpha(1 - \alpha)\|\mathscr{N}(u^k - \tilde{u}^k)\|^2. \end{aligned}$$

The assertions are proved. □

Moreover, to further alleviate the computation, the proposed backward substitution procedure (49a) is eligible to simply take a constant

$$\alpha_k \equiv \mu \in (0,1)$$

as the stepsize. This is, in fact, a by-product of (55). From (49b) and (56), we have

$$\alpha_k^B \geq \tfrac{1}{2}. \tag{57}$$

Therefore we have $\gamma \cdot \alpha_k^B \geq \gamma \cdot \frac{1}{2} \in (0,1)$. If we take a constant $\mu \in (0,1)$ as the stepsize for the backward substitution procedure (49a), instead of (49b), the convergence is still ensured. Again, although it saves computation by taking a constant stepsize in $(0,1)$, we also have observed some applications where the stepsize chosen by (49b) can accelerate the convergence of Algorithm 3. Thus, similarly as Algorithm 2, to implement Algorithm 3, our recommendation is to use a constant stepsize if the computation of (49b) is computationally expensive; or just use (49b) otherwise.

Now, we can show that the sequence $\{u^k\}$ generated by the proposed Algorithm 3 is contractive with respect to the set $\mathcal{U}^*$ under the $(\mathcal{P}\mathcal{P}^T)$-norm. Based on this fact, the reason why γ should be restricted into the interval $(0,2)$ is also clear.

Theorem 5.9. *Let the sequence $\{u^k\}$ be generated by the proposed Algorithm 3. Then, we have*

$$\|u^{k+1} - u^*\|_G^2 \leq \|u^k - u^*\|_G^2 - \frac{\gamma(2-\gamma)}{4}\|u^k - \tilde{u}^k\|^2, \quad \forall\, u^* \in \mathcal{U}^*. \tag{58}$$

Proof. Since $\alpha_k = \gamma\alpha_k^B$, it follows from (51) and $\alpha_k^B \geq \frac{1}{2}$ that

$$\begin{aligned} q^B(\alpha_k) &= \gamma\alpha_k^B(\|u^k - \tilde{u}^k\|^2 + \|\mathcal{F}(u^k - \tilde{u}^k)\|^2) - \gamma^2\alpha_k^B(\alpha_k^B\|\mathcal{N}(u^k - \tilde{u}^k)\|^2) \\ &\overset{(49b)}{=} \gamma\left(1 - \frac{\gamma}{2}\right)\alpha_k^B(\|u^k - \tilde{u}^k\|^2 + \|\mathcal{F}(u^k - \tilde{u}^k)\|^2) \\ &\geq \frac{\gamma(2-\gamma)}{4}\|u^k - \tilde{u}^k\|^2. \end{aligned}$$

The assertion follows from the above inequality and (50) directly. □

Remark 5.10. Let us revisit the case of (1) with $m = 2$. For this special case, it follows from (13) and (14) that $\mathcal{P} = \mathcal{I}$ and $\mathcal{L} = \mathcal{N}$. We thus have $q^F(\alpha) = q^B(\alpha)$ (see (40) and (51)). In Subsection 5.1, we have demonstrated that if the stepsize is taken as $\alpha \equiv 1$, Algorithm 2 reduces to the standard ADMM scheme (2). Similarly, if $\alpha \equiv 1$, the proposed backward substitution procedure reduces to

$$\mathcal{P}^T(u^{k+1} - u^k) = \mathcal{N}(\tilde{u}^k - u^k).$$

Because $\mathcal{P}$ is the identity matrix and $\mathcal{L} = \mathcal{N}$, Algorithm 3 also reduces to the standard ADMM scheme (2). In short, for the case $m = 2$ in (1), the proposed forward and backward substitutions are identical, and Algorithms 2 and 3 reduce to the standard ADMM scheme (2). Our proposed algorithms are thus able to recover the standard ADMM scheme (2) when $m = 2$ in (1).

5.3. Convergence Analysis

In the last subsections, we have shown that the sequences generated by Algorithms 2 and 3 are contractive with respective to the set $\mathcal{U}^*$. This fact thus enables us to establish the global convergence for both algorithms simultaneously based on the analytic framework of contraction methods in Blum and Oettli [1]. Moreover, from the contraction perspective, the local linear convergence of Algorithms 2 and 3 can be derived immediately provided that certain error bounds (e.g., some analogous to those in Hong and Luo [24]) are assumed to be satisfied. As we have mentioned, the input of the proposed algorithms at each iteration is u^k. Thus, in the following analysis, we investigate the convergence for the sequence $\{u^k\}$.

In fact, the inequalities (42) and (58) in which the contraction of the sequences generated by Algorithms 2 and 3 are shown, can be unified as

$$\|u^{k+1} - u^*\|_G^2 \leq \|u^k - u^*\|_G^2 - c \cdot \|u^k - \tilde{u}^k\|^2, \quad \forall\, u^* \in \mathcal{U}^*, \tag{59}$$

where $c > 0$ is certain constant. More precisely, $G = \mathcal{I}$ and $c = \gamma(2-\gamma)/(4\|\mathcal{L}^T\mathcal{L}\|)$ for Algorithm 2; and $G = \mathcal{P}\mathcal{P}^T$ and $c = \gamma(2-\gamma)/4$ for Algorithm 3.

Theorem 5.11 (Global Convergence). *Let the sequence $\{u^k\}$ be generated by the proposed framework of ADMM with a substitution (either Algorithm 2 or Algorithm 3). Then, there exists $u^\infty \in \mathcal{U}^*$ such that*

$$\lim_{k\to\infty} u^k = u^\infty.$$

Proof. First, for an arbitrarily fixed $u^* \in \mathcal{U}^*$, it follows from (59) that the sequence $\{u^k\}$ is bounded. Summarizing the inequality (59) over $k = 0, 1, \ldots$, we obtain

$$\sum_{k=0}^{\infty} c \cdot \|u^k - \tilde{u}^k\|^2 \le \|u^0 - u^*\|_G^2,$$

and thus

$$\lim_{k\to\infty} \|u^k - \tilde{u}^k\|^2 = 0. \tag{60}$$

Therefore the sequence $\{\tilde{u}^k\}$ is also bounded. Since A_j's are assumed to be full column rank, it follows from (20b) that $\{\tilde{w}^k\}$ is bounded. Then, there exists a cluster point w^∞ and a subsequence $\{\tilde{w}^{k_l}\}$ of $\{\tilde{w}^k\}$ converging to w^∞. Recall the notation in (7b) and (8). We specify $x^\infty, w^\infty, u^\infty$, respectively, as

$$x^\infty = \begin{pmatrix} x_1^\infty \\ \vdots \\ x_m^\infty \end{pmatrix}, \qquad w^\infty = \begin{pmatrix} x_1^\infty \\ \vdots \\ x_m^\infty \\ \lambda^\infty \end{pmatrix}, \qquad u^\infty = \begin{pmatrix} H^{1/2}A_2x_2^\infty \\ \vdots \\ H^{1/2}A_mx_m^\infty \\ H^{-1/2}\lambda^\infty \end{pmatrix}. \tag{61}$$

In addition, we have

$$\lim_{l\to\infty} \|\tilde{x}_j^{k_l} - x_j^\infty\| = 0, \quad j = 1, \ldots, m. \tag{62}$$

Since H is assumed to be positive definite, it follows that $\lim_{l\to\infty} \|\tilde{u}^{k_l} - u^\infty\| = 0$. Combining this with (60) and the positive definiteness of G, we immediately have

$$\lim_{l\to\infty} \|u^{k_l} - u^\infty\|_G = 0. \tag{63}$$

On the other hand, because of the continuity of θ, taking limit in (22) along the subsequence $\{\tilde{w}^{k_l}\}$ and using (60) again, we obtain

$$\theta(x) - \theta(x^\infty) + \begin{pmatrix} x_1 - x_1^\infty \\ x_2 - x_2^\infty \\ \cdots \\ x_i - x_i^\infty \\ \cdots \\ x_m - x_m^\infty \end{pmatrix}^T \begin{pmatrix} -A_1^T\lambda^\infty \\ -A_2^T\lambda^\infty \\ \cdots \\ -A_i^T\lambda^\infty \\ \cdots \\ -A_m^T\lambda^\infty \end{pmatrix} \ge 0, \quad \forall\, x \in \mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_m. \tag{64}$$

Note that

$$\begin{aligned} \left\| \sum_{j=1}^m A_j x_j^\infty - b \right\| &\le \left\| \sum_{j=1}^m A_j \tilde{x}_j^k - b \right\| + \left\| \sum_{j=1}^m A_j \tilde{x}_j^k - \sum_{j=1}^m A_j x_j^\infty \right\| \\ &\le \left\| \sum_{j=1}^m A_j \tilde{x}_j^k - b \right\| + \sum_{j=1}^m \|A_j\| \cdot \|\tilde{x}_j^k - x_j^\infty\|. \end{aligned} \tag{65}$$

It follows from (23) that

$$\begin{aligned} \left\| \sum_{j=1}^m A_j \tilde{x}_j^k - b \right\| &\le \|H^{-1}(\tilde{\lambda}^k - \lambda^k)\| + \left\| \sum_{j=2}^m (A_j \tilde{x}_j^k - A_j x_j^k) \right\| \\ &\le \|H^{-1}\| \cdot \|(\tilde{\lambda}^k - \lambda^k)\| + \sum_{j=2}^m \|A_j\| \cdot \|\tilde{x}_j^k - x_j^k\|. \end{aligned} \tag{66}$$

Also, because of (60), we have

$$\lim_{k\to\infty} \|\lambda^k - \tilde{\lambda}^k\| = 0 \qquad \text{and} \qquad \lim_{k\to\infty} \|\tilde{x}_j^k - x_j^k\| = 0, \quad \forall\, j = 2, \ldots, m. \tag{67}$$

Based on (62), (66), and (67), we take limit in (65) along the subsequence $\{\tilde{w}^{k_l}\}$ and obtain

$$\sum_{j=1}^{m} A_j x_j^{\infty} = b. \tag{68}$$

Recall the VI reformulation (7). Then, it follows from (64) and (68) that

$$\theta(x) - \theta(x^{\infty}) + (w - w^{\infty})^T F(w^{\infty}) \geq 0, \quad \forall\, w \in \mathcal{W}, \tag{69}$$

which means w^{∞} is a solution point of VI($\mathcal{W}, F, \theta$). From (61), this also means $u^{\infty} \in \mathcal{U}^*$. Furthermore, it follows from (59) and (63) that $\lim_{k\to\infty} \|u^k - u^{\infty}\|_G^2 = 0$ and thus $\lim_{k\to\infty} u^k = u^{\infty}$. The proof is complete. □

Remark 5.12. Since the matrices A_i's are assumed to be full column rank, the convergence of $\{x_2^k, \ldots, x_m^k, \lambda^k\}$ follows from Theorem 5.11 immediately. We refer to He et al. [20], Zhang et al. [47] for how to derive the convergence of $\{x_2^k, \ldots, x_m^k, \lambda^k\}$ from Theorem 5.11 without the full column rank assumption on the coefficient matrices A_i's.

Now, we can show the local linear convergence of Algorithms 2 and 3 immediately under some error bound assumptions similar as those in Hong and Luo [24].

Theorem 5.13 (Local Linear Convergence Rate)**.** *Let the sequence $\{u^k\}$ be generated by the proposed framework of ADMM with a substitution (either Algorithm 2 or Algorithm 3). Assume that there is a constant $\tau > 0$ such that*

$$\|u^k - u^*\| \leq \tau \|u^k - \tilde{u}^k\|, \; u^* \in \mathcal{U}^*. \tag{70}$$

Then, $\{u^k\}$ converges to u^* on a linear rate.

Proof. Since $\{\|u^k - u^*\|_G\}$ is Fejér monotone with respect to $\mathcal{U}^*$, the sequence $\{u^k\}$ is bounded. Then, the local linear convergence rate of $\{u^k\}$ is an immediate assertion based on the assumption (70) and the fact (59). □

6. Worst-Case Convergence Rate in the Nonergodic Sense

In this section, our purpose is to show that without the error bound assumption in Theorem 5.13, we still can estimate the worst-case convergence rates measured by the iteration complexity in the nonergodic sense for Algorithms 2 and 3. The basis of the analysis in this section is the fact that the assertion (21) in Theorem 3.3 can be rewritten as

$$\tilde{w}^k \in \mathcal{W}, \qquad \theta(x) - \theta(\tilde{x}^k) + (w - \tilde{w}^k)^T F(\tilde{w}^k) + (u - \tilde{u}^k)^T \mathcal{L}(\tilde{u}^k - u^k) \geq 0, \quad \forall\, w \in \mathcal{W}.$$

Since $\mathcal{L}$ is a nonsingular matrix, the above inequality means that $\tilde{w}^k$ is a solution point in $\mathcal{W}^*$ if $\|u^k - \tilde{u}^k\|^2 = 0$. Thus we can view $\|u^k - \tilde{u}^k\|^2$ as a residual or an error bound to measure the accuracy of $\tilde{w}^k$ to a solution point in $\mathcal{W}^*$. In this section, we will show that after t iterations of the proposed Algorithms 2 and 3, we can ensure that

$$\min_{0\leq k\leq t}\{\|u^k - \tilde{u}^k\|^2\} \leq \epsilon \qquad \text{or} \qquad \|u^t - \tilde{u}^t\|^2 \leq \epsilon,$$

where $\epsilon = \mathbf{O}(\mathbf{1/t})$. Thus, a worst-case $\mathbf{O}(\mathbf{1/t})$ convergence rate in the nonergodic sense is established for Algorithms 2 and 3.

We have shown that the stepsize for either the forward or backward substitution step (i.e., (37a) or (49a)) can be simply taken as a constant in certain intervals. This could be useful for further accelerating Algorithms 2 and 3 for the case where computing the stepsize via (37b) or (49b) is expensive. To show the worst-case $\mathbf{O}(\mathbf{1/t})$ convergence rate in the nonergodic sense for Algorithms 2 and 3, we consider the cases with a constant and chosen stepsize, respectively.

6.1. The Case with a Chosen Stepsize

We first establish a worst-case $\mathbf{O}(\mathbf{1/t})$ convergence rate in the nonergodic sense for Algorithms 2 and 3 when their substitution stepsizes are taken as (37b) and (49b)), respectively.

For both algorithms, it follows from (59) that

$$c\sum_{k=0}^{\infty} \|u^k - \tilde{u}^k\|^2 \leq \|u^0 - u^*\|_G^2, \quad \forall\, u^* \in \mathcal{U}^*,$$

where $c = \gamma(2-\gamma)/(4\|\mathscr{L}^T\mathscr{L}\|)$ and $G = \mathscr{I}$ for Algorithm 2; and $c = \gamma(2-\gamma)/4$ and $G = \mathscr{P}\mathscr{P}^T$ for Algorithm 3. Thus, for any integer $t > 0$, we obtain

$$c\sum_{k=0}^{t}\|u^k - \tilde{u}^k\|^2 \le \|u^0 - u^*\|_G^2, \quad \forall\, u^* \in \mathscr{U}^*,$$

and consequently, it follows that

$$\min_{0\le k\le t}\{\|u^k - \tilde{u}^k\|^2\} \le \frac{1}{c(t+1)}\|u^0 - u^*\|_G^2, \quad \forall\, u^* \in \mathscr{U}^*. \tag{71}$$

Recall $\mathscr{W}^*$ is convex and closed under our assumptions (see Theorem 2.3.5 in Facchinei and Pang [10]). Let

$$d := \inf\{\|u^0 - u^*\|_G^2 \mid u^* \in \mathscr{U}^*\}.$$

For any given $\epsilon > 0$, the inequality (71) indicates that Algorithms 2 and 3 require at most $\lfloor d/(c\epsilon)\rfloor$ iterations to fulfill the requirement $\|u^k - \tilde{u}^k\|^2 \le \epsilon$. Thus, a worst-case $\mathbf{O(1/t)}$ convergence rate is established for Algorithms 2 and 3 in the nonergodic sense.

6.2. The Case with a Constant Stepsize

Then, we consider the case with a constant stepsize. We first show a lemma where an inequality regarding the output of the ADMM procedure (20) is proved.

Lemma 6.1. *Let $\tilde{w}^k$ be generated by the ADMM procedure (20) with given u^k and $\tilde{u}^k = \mathscr{A}\tilde{v}^k$. Then, we have*

$$(u^k - u^{k+1})^T\mathscr{L}((u^k - \tilde{u}^k) - (u^{k+1} - \tilde{u}^{k+1})) \ge \tfrac{1}{2}\|(u^k - \tilde{u}^k) - (u^{k+1} - \tilde{u}^{k+1})\|^2_{(\mathscr{L}^T+\mathscr{L})}. \tag{72}$$

Proof. First, it follows from (21) that

$$\tilde{w}^k \in \mathscr{W}, \qquad \theta(x) - \theta(\tilde{x}^k) + (w - \tilde{w}^k)^T F(\tilde{w}^k) \ge (u - \tilde{u}^k)\mathscr{L}(u^k - \tilde{u}^k), \quad \forall\, w \in \mathscr{W}. \tag{73}$$

This inequality is also true for $k := k+1$. Thus we have

$$\tilde{w}^{k+1} \in \mathscr{W}, \qquad \theta(x) - \theta(\tilde{x}^{k+1}) + (w - \tilde{w}^{k+1})^T F(\tilde{w}^{k+1}) \ge (u - \tilde{u}^{k+1})\mathscr{L}(u^{k+1} - \tilde{u}^{k+1}), \quad \forall\, w \in \mathscr{W}. \tag{74}$$

Setting $w = \tilde{w}^{k+1}$ and $w = \tilde{w}^k$ in (73) and (74), respectively, and then adding these two resulting inequalities, we obtain

$$(\tilde{u}^k - \tilde{u}^{k+1})^T\mathscr{L}((u^k - \tilde{u}^k) - (u^{k+1} - \tilde{u}^{k+1})) \ge (\tilde{w}^k - \tilde{w}^{k+1})^T(F(\tilde{w}^k) - F(\tilde{w}^{k+1})).$$

Using the monotonicity of F, we have

$$(\tilde{u}^k - \tilde{u}^{k+1})^T\mathscr{L}((u^k - \tilde{u}^k) - (u^{k+1} - \tilde{u}^{k+1})) \ge 0. \tag{75}$$

Adding the term

$$((u^k - \tilde{u}^k) - (u^{k+1} - \tilde{u}^{k+1}))^T\mathscr{L}((u^k - \tilde{u}^k) - (u^{k+1} - \tilde{u}^{k+1}))$$

to both sides of (75) and by a simple manipulation, we get (72) and the lemma is proved. □

In the following, we analyze the worst-case $\mathbf{O(1/t)}$ convergence rate in the nonergodic sense for Algorithms 2 and 3 where their substitution stepsizes are constant.

6.2.1. Algorithm 2 with a Constant Stepsize. Recall that the stepsize in the substitution step of Algorithm 2 can be taken as a constant in $\alpha \in (0, \alpha_*^F)$, where α_*^F is defined in (45). We first show that the sequence $\{\|\mathscr{L}(u^k - \tilde{u}^k)\|^2\}$ is monotonically nonincreasing and then derive the worst-case $\mathbf{O(1/\epsilon)}$ convergence rate.

Lemma 6.2. *Let $\{u^k\}$ be generated by Algorithm 2 with $\alpha_k \equiv \alpha \in (0, \alpha_*^F)$. Then, we have*

$$\|\mathscr{L}(u^{k+1} - \tilde{u}^{k+1})\|^2 \le \|\mathscr{L}(u^k - \tilde{u}^k)\|^2, \quad \forall\, k \ge 0. \tag{76}$$

Proof. First, using (49a), we have

$$u^k - u^{k+1} = \alpha \mathcal{L}(u^k - \tilde{u}^k).$$

Then, substituting it into (72), we get

$$(u^k - \tilde{u}^k)^T \mathcal{L}^T \mathcal{L}((u^k - \tilde{u}^k) - (u^{k+1} - \tilde{u}^{k+1})) \geq \frac{1}{2\alpha} \|(u^k - \tilde{u}^k) - (u^{k+1} - \tilde{u}^{k+1})\|_{(\mathcal{L}^T + \mathcal{L})}. \tag{77}$$

Setting $a = \mathcal{L}(u^k - \tilde{u}^k)$ and $b = \mathcal{L}(u^{k+1} - \tilde{u}^{k+1})$ in the identity

$$\|a\|^2 - \|b\|^2 = 2a^T(a - b) - \|a - b\|^2,$$

and using the inequality (77) and the relationship (18), we obtain

$$\begin{aligned} \|\mathcal{L}(u^k - \tilde{u}^k)\|^2 - \|\mathcal{L}(u^{k+1} - \tilde{u}^{k+1})\|^2 &\geq \frac{1}{\alpha} \|(u^k - \tilde{u}^k) - (u^{k+1} - \tilde{u}^{k+1})\|_{(\mathcal{L}^T + \mathcal{L})} - \|\mathcal{L}(u^k - \tilde{u}^k) - \mathcal{L}(u^{k+1} - \tilde{u}^{k+1})\|^2 \\ &= \frac{1}{\alpha} \|(u^k - \tilde{u}^k) - (u^{k+1} - \tilde{u}^{k+1})\|^2_{((\mathcal{I} + \mathcal{S}^T\mathcal{S}) - \alpha\mathcal{L}^T\mathcal{L})}. \end{aligned} \tag{78}$$

Because $\alpha \in (0, \alpha_*^F)$ and thus $\mathcal{I} + \mathcal{S}^T\mathcal{S} - \alpha\mathcal{L}^T\mathcal{L} \succeq 0$ (see (45)), the right-hand side of (78) is nonnegative and the lemma is proved. □

Now, we are ready to estimate a worst-case $\mathbf{O(1/t)}$ convergence rate in the nonergodic sense for Algorithm 2 with a constant substitution stepsize.

Theorem 6.3. *Let $\{u^t\}$ be the sequence generated by the proposed Algorithm 2 with $\alpha_k \equiv \alpha \in (0, \alpha_*^F)$. Then, we have*

$$\|\mathcal{L}(u^t - \tilde{u}^t)\|^2 \leq \frac{1}{\alpha(\alpha_*^F - \alpha)(t+1)} \|u^0 - u^*\|^2, \tag{79}$$

$\mathcal{L}$ is defined in (14).

Proof. First, it follows from Theorem 5.6 (see (47)) that

$$\alpha(\alpha_*^F - \alpha) \sum_{k=0}^{\infty} \|\mathcal{L}(u^k - \tilde{u}^k)\|^2 \leq \|u^0 - u^*\|^2, \quad \forall\, u^* \in \mathcal{U}^*. \tag{80}$$

Note that Lemma 6.2 shows that the sequence $\{\|\mathcal{L}(u^k - \tilde{u}^k)\|^2\}$ is monotonically nonincreasing. Thus we have

$$(t+1)\|\mathcal{L}(u^t - \tilde{u}^t)\|^2 \leq \sum_{k=0}^{t} \|\mathcal{L}(u^k - \tilde{u}^k)\|^2, \tag{81}$$

which implies the assertion (79) immediately. □

Recall $\mathcal{W}^*$ is convex and closed under our assumptions (see Facchinei and Pang [10, Theorem 2.3.5]). Let

$$d_1 := \inf\{\|u^0 - u^*\|^2 \mid u^* \in \mathcal{U}^*\}.$$

For any given $\epsilon > 0$, Theorem 6.3 indicates that Algorithm 2 with $\alpha_k \equiv \alpha \in (0, \alpha_*^F)$ requires at most $\lfloor d_1/(\alpha(\alpha_*^F - \alpha)\epsilon) \rfloor$ iterations to ensure that $\|\mathcal{L}(u^k - \tilde{u}^k)\|^2 \leq \epsilon$. A worst-case $\mathbf{O(1/t)}$ convergence rate in the nonergodic sense is thus established for Algorithm 2 with $\alpha_k \equiv \alpha \in (0, \alpha_*^F)$.

6.2.2. Algorithm 3 with a Constant Stepsize. Recall that the stepsize in the substitution step of Algorithm 3 can be taken as a constant in $\alpha \in (0, 1)$. First, it follows from (50) and (54) that

$$\|u^{k+1} - u^*\|_G^2 \leq \|u^k - u^*\|_G^2 - \alpha(1 - \alpha)\|\mathcal{N}(u^k - \tilde{u}^k)\|^2, \quad \forall\, u^* \in \mathcal{U}^*. \tag{82}$$

Then, we need to show that the sequence $\{\|\mathcal{N}(u^k - \tilde{u}^k)\|^2\}$ is monotonically nonincreasing before we derive the worst-case $\mathbf{O(1/t)}$ convergence rate.

Lemma 6.4. *Let $\{u^k\}$ be generated by Algorithm 3 with $\alpha_k \equiv \mu \in (0, 1)$. Then, we have*

$$\|\mathcal{N}(u^{k+1} - \tilde{u}^{k+1})\|^2 \leq \|\mathcal{N}(u^k - \tilde{u}^k)\|^2, \quad \forall\, k \geq 0. \tag{83}$$

Proof. Using (49a), we have

$$u^k - u^{k+1} = \alpha \mathscr{P}^{-T} \mathscr{N}(u^k - \tilde{u}^k).$$

Substituting it into (72) and using $\mathscr{N} = \mathscr{P}^{-1}\mathscr{L}$, we get

$$(u^k - \tilde{u}^k)^T \mathscr{N}^T \mathscr{N}((u^k - \tilde{u}^k) - (u^{k+1} - \tilde{u}^{k+1})) \geq \frac{1}{2\alpha} \|(u^k - \tilde{u}^k) - (u^{k+1} - \tilde{u}^{k+1})\|_{(\mathscr{L}^T + \mathscr{L})}. \tag{84}$$

In addition, setting $a = \mathscr{N}(u^k - \tilde{u}^k)$ and $b = \mathscr{N}(u^{k+1} - \tilde{u}^{k+1})$ in the identity

$$\|a\|^2 - \|b\|^2 = 2a^T(a - b) - \|a - b\|^2,$$

and using the inequality (84), we obtain

$$\begin{aligned} \|\mathscr{N}(u^k - \tilde{u}^k)\|^2 - \|\mathscr{N}(u^{k+1} - \tilde{u}^{k+1})\|^2 &\geq \frac{1}{\alpha} \|(u^k - \tilde{u}^k) - (u^{k+1} - \tilde{u}^{k+1})\|_{(\mathscr{L}^T + \mathscr{L})} - \|\mathscr{N}(u^k - \tilde{u}^k) - \mathscr{N}(u^{k+1} - \tilde{u}^{k+1})\|^2 \\ &\geq \|(u^k - \tilde{u}^k) - (u^{k+1} - \tilde{u}^{k+1})\|_{(\mathscr{L}^T + \mathscr{L})} - \|(u^k - \tilde{u}^k) - (u^{k+1} - \tilde{u}^{k+1})\|^2_{(\mathscr{N}^T \mathscr{N})}. \end{aligned} \tag{85}$$

The last inequality is due to $\alpha \in (0, 1]$. It follows from (13) and (14) that

$$\mathscr{L}^T + \mathscr{L} = \begin{pmatrix} 2I_l & I_l & \cdots & I_l & -I_l \\ I_l & 2I_l & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & I_l & -I_l \\ I_l & \cdots & I_l & 2I_l & -I_l \\ -I_l & \cdots & -I_l & -I_l & 2I_l \end{pmatrix} \quad \text{and} \quad \mathscr{N}^T \mathscr{N} = \begin{pmatrix} 2I_l & I_l & \cdots & I_l & -I_l \\ I_l & 2I_l & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & I_l & -I_l \\ I_l & \cdots & I_l & 2I_l & -I_l \\ -I_l & \cdots & -I_l & -I_l & I_l \end{pmatrix}.$$

Thus the right-hand side of (85) is nonnegative and the lemma is proved. □

Now, we are ready to estimate a worst-case $\mathbf{O(1/t)}$ convergence rate in the nonergodic sense for Algorithm 3 with a constant substitution stepsize.

Theorem 6.5. *Let $\{u^t\}$ be the sequence generated by the proposed Algorithm 3 with $\alpha_k \equiv \mu \in (0, 1)$. Then, we have*

$$\|\mathscr{N}(u^t - \tilde{u}^t)\|^2 \leq \frac{1}{\alpha(1 - \alpha)(t + 1)} \|u^0 - u^*\|^2_G, \tag{86}$$

where $G = \mathscr{P}\mathscr{P}^T$ and $\mathscr{P}$ is defined in (14).

Proof. First, it follows from (82) that

$$\alpha(1 - \alpha) \sum_{k=0}^{\infty} \|\mathscr{N}(u^k - \tilde{u}^k)\|^2 \leq \|u^0 - u^*\|^2_G, \quad \forall\, u^* \in \mathscr{U}^*. \tag{87}$$

Note that Lemma 6.4 shows that the sequence $\{\|\mathscr{N}(u^k - \tilde{u}^k)\|^2\}$ is monotonically nonincreasing. Thus we have

$$(t + 1)\|\mathscr{N}(u^t - \tilde{u}^t)\|^2 \leq \sum_{k=0}^{t} \|\mathscr{N}(u^k - \tilde{u}^k)\|^2, \tag{88}$$

which implies the assertion (86) immediately. □

Recall $\mathscr{W}^*$ is convex and closed under our assumptions (see Facchinei and Pang [10, Theorem 2.3.5]). Let

$$d_2 := \inf\{\|u^0 - u^*\|^2_G \mid u^* \in \mathscr{U}^*\}.$$

For any given $\epsilon > 0$, Theorem 6.5 indicates that Algorithm 3 with $\alpha_k \equiv \mu \in (0, 1)$ requires at most $\lfloor d_2/(\mu(1 - \mu)\epsilon) \rfloor$ iterations to ensure that $\|\mathscr{N}(u^k - \tilde{u}^k)\|^2 \leq \epsilon$. A worst-case $\mathbf{O(1/t)}$ convergence rate in the nonergodic sense is thus established for Algorithm 3 with $\alpha_k \equiv \mu \in (0, 1)$.

7. Numerical Experiments

In this section, we apply the proposed Algorithms 2 and 3 (denoted by "ADMM-Forward" and "ADMM-Backward," respectively) to solve some specific applications of the model (1) arising in image processing and report the numerical results. For "ADMM-Forward" and "ADMM-Backward," we will test both cases where the stepsize is chosen dynamically by the proposed strategies and fixed as a constant. We also compare them numerically with the direct extension of ADMM (3) (denoted by "EADMM") and the ADMM with Gaussian back substitution in He et al. [19] (denoted by "ADMM-GB").

Note that all the methods to be tested involve an ADMM procedure (3) or (20). For the penalty matrix H in (3) or (20), we choose it as a block diagonal matrix in the following form for all the methods:

$$H = \begin{pmatrix} \beta_1^2 I_{n_1} & 0 & \dots & 0 \\ 0 & \beta_2^2 I_{n_2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \beta_m^2 I_{n_m} \end{pmatrix}, \tag{89}$$

where $\beta_i > 0$ and I_{n_i} is the identity matrix in $\Re^{n_i \times n_i}$ for $i = 1, 2, \dots, m$.

All the methods were coded by MATLAB 7.1, and all numerical experiments were performed on a personal Lenovo laptop computer with Intel Core i5-6300U processor @ 2.30 GHz and 8 GB memory.

7.1. Example 1—An Image Restoration Model for Mixed Noise Removal

Let $\mathbf{x} \in \Re^n$ represent a digital image with $n = l_1 \times l_2$. Note that a two-dimensional image can be represented by vectorizing it as a one-dimensional vector in certain order, e.g., the lexicographic order. We consider the following image restoration model for mixed noise removal in Huang et al. [25]:

$$\min_{\mathbf{x},\mathbf{y}} \left\{ \tau \| |\nabla \mathbf{x}| \|_1 + \frac{\rho}{2} \|\mathbf{x} - \mathbf{y}\|^2 + \|P_\Omega(B\mathbf{y} - \mathbf{x}^0)\|_1 \right\}, \tag{90}$$

where $\|\cdot\|_1$ and $\|\cdot\|$ denote the l_1 and l_2 norms, respectively; $\nabla = (\partial_1, \partial_2)$ denotes the gradient operator and $\| |\nabla \cdot| \|_1$ is the total variation term (see, e.g., Rudin et al. [38]), which can induce sparse representation of a piecewise smooth function; B is the spatially invariant convolutional matrix generating blur to the image; Ω represents the set of pixels, which are corrupted by the impulsive noise (all the pixels outside Ω are corrupted by the Gaussian noise); P_Ω is the characteristic function of the set Ω, i.e., $P_\Omega(\mathbf{x})$ has the value 1 for any pixel within Ω and 0 for any pixel outside Ω; $\mathbf{x}^0$ is the corrupted image with blurry and mixed noise; and τ and ρ are positive constants.

We first show that the model (90) can be reformulated as a special case of (1). In fact, by introducing the auxiliary variables $\mathbf{u}$, $\mathbf{v}$, and $\mathbf{z}$, we can reformulate (90) as

$$\begin{array}{ll} \min & \tau \| |\mathbf{u}| \|_1 + \frac{\rho}{2} \|\mathbf{v}\|^2 + \|P_\Omega(\mathbf{z})\|_1 \\ \text{s.t.} & \mathbf{u} = \nabla \mathbf{x}, \\ & \mathbf{v} = \mathbf{x} - \mathbf{y}, \\ & \mathbf{z} = B\mathbf{y} - \mathbf{x}^0, \end{array} \tag{91}$$

which is a special case of the abstract model (1) with the following specification:

- $x_1 := \mathbf{x}$, $x_2 := \mathbf{y}$ and $x_3 := (\mathbf{u}, \mathbf{v}, \mathbf{z})$; $\mathscr{X}_i$ $(i = 1, 2, 3)$ are the whole real spaces in appropriate dimensionality;
- $\theta_1(x_1) := 0$, $\theta_2(x_2) := 0$ and $\theta_3(x_3) := \theta_3(\mathbf{u}, \mathbf{v}, \mathbf{z}) = \tau \| |\mathbf{u}| \|_1 + (\rho/2)\|\mathbf{v}\|^2 + \|P_\Omega(\mathbf{z})\|_1$;
- and

$$A_1 := \begin{bmatrix} \nabla \\ I \\ 0 \end{bmatrix}, \quad A_2 := \begin{bmatrix} 0 \\ -I \\ B \end{bmatrix}, \quad A_3 := \begin{bmatrix} -I & 0 & 0 \\ 0 & -I & 0 \\ 0 & 0 & -I \end{bmatrix}, \quad b := \begin{bmatrix} 0 \\ 0 \\ \mathbf{x}^0 \end{bmatrix}. \tag{92}$$

Thus the methods "EADMM," "ADMM-GB," "ADMM-Forward," and "ADMM-Backward" are all applicable to the model (91). Below we elaborate on the minimization subproblems arising in the ADMM procedure (3) or (20) and show that they all have closed-form solutions.

- The $\tilde{x}_1$-subproblem in (20), i.e., the $\tilde{\mathbf{x}}$-subproblem for (91), can be formulated as

$$\begin{aligned} \tilde{x}_1^k &= \arg\min_{x_1} \{ \|A_1 x_1 + H^{-1/2}(u_2^k + u_3^k) - b - u_\lambda^k\|_H^2 \} \\ &\Leftrightarrow \quad (\beta_1^2 \nabla^T \nabla + \beta_2^2 I)\tilde{x}_1^k = \beta_1 \nabla^T (u_{21}^k + u_{31}^k - u_{\lambda 1}^k) - \beta_2 (u_{22}^k + u_{32}^k - u_{\lambda 2}^k), \end{aligned} \tag{93}$$

which can be solved efficiently by the fast Fourier transform (FFT) or the discrete cosine transform (DCT) (see, e.g., Hansen et al. [18] for details).

- The $\tilde{x}_2$-subproblem in (20), i.e., the $\tilde{\mathbf{y}}$-subproblem for (91), can be written as

$$\begin{aligned}\tilde{x}_2^k &= \arg\min_{x_2}\{\|A_1\tilde{x}_1^k + A_2x_2 + H^{-1/2}u_3^k - b - u_\lambda^k\|_H^2\}\\ &\Leftrightarrow \quad (\beta_2^2 I + \beta_3^2 B^T B)\tilde{x}_2^k = \beta_3 B^T(\beta_3\mathbf{x}_0 - u_{33}^k + u_{\lambda 3}^k) + \beta_2(u_{32}^k - u_{\lambda 2}^k + \beta_2\tilde{\mathbf{x}}^k),\end{aligned}$$

which can also be solved by the fast transforms as that of (93).

- The $\tilde{x}_3$-subproblem in (20), i.e., the $(\tilde{\mathbf{u}}, \tilde{\mathbf{v}}, \tilde{\mathbf{z}})$-subproblem for (91), reads as

$$\begin{aligned}\tilde{x}_3^k = (\tilde{\mathbf{u}}^k, \tilde{\mathbf{v}}^k, \tilde{\mathbf{z}}^k) = \arg\min_{\mathbf{u},\mathbf{v},\mathbf{z}}\Bigg\{&\tau\|\|\mathbf{u}\|\|_1 + \frac{\rho}{2}\|\mathbf{v}\|^2 + \|P_\Omega(\mathbf{z})\|_1 + \frac{\beta_1^2}{2}\left\|\nabla\tilde{\mathbf{x}}^k - \mathbf{u} - \frac{u_{\lambda 1}^k}{\beta_1}\right\|^2\\ &+ \frac{\beta_2^2}{2}\left\|\tilde{\mathbf{x}}^k - \tilde{\mathbf{y}}^k - \mathbf{v} - \frac{u_{\lambda 2}^k}{\beta_2}\right\|^2 + \frac{\beta_3^2}{2}\left\|B\tilde{\mathbf{y}}^k - \mathbf{z} - \mathbf{x}^0 - \frac{u_{\lambda 3}^k}{\beta_3}\right\|^2\Bigg\},\end{aligned}$$

and it can be solved separably as follows:

—$\tilde{\mathbf{u}}^k = \text{shrink}_{\tau/\beta_1^2}(\nabla\tilde{\mathbf{x}}^k - u_{\lambda 1}^k/\beta_1)$, where $\text{shrink}_\sigma(\cdot)$ denotes the well-known shrinkage operator (see, e.g., Donoho [7]). That is,

$$\text{shrink}_\sigma(a) = \text{sign}(a)\circ\max\{|a| - \sigma, 0\}, \quad \forall\, a \in \Re^n,$$

with $\sigma > 0$, where $\text{sign}(\cdot)$ is the sign function and the operator "$\circ$" stands for the componentwise scalar multiplication.

—$\tilde{\mathbf{v}}^k = \beta_2[\beta_2(\tilde{\mathbf{x}}^k - \tilde{\mathbf{y}}^k) - u_{\lambda 2}^k]/(\rho + \beta_2^2)$.

—$\tilde{\mathbf{z}}^k$ is given by

$$\tilde{\mathbf{z}}^k = \arg\min_{\mathbf{z}}\left\{\|P_\Omega(\mathbf{z})\|_1 + \frac{\beta_3^2}{2}\left\|B\tilde{\mathbf{y}}^k - \mathbf{z} - \mathbf{x}^0 - \frac{u_{\lambda 3}^k}{\beta_3}\right\|^2\right\},$$

whose closed-form solution can be obtained via

$$(\tilde{\mathbf{z}}^k)_i = \begin{cases}[\text{shrink}_{(1/\beta_3^2)}(B\tilde{\mathbf{y}}^k - \mathbf{x}^0 - u_{\lambda 3}^k/\beta_3)]_i, & \text{if } i \in \Omega,\\ [B\tilde{\mathbf{y}}^k - \mathbf{x}^0 - u_{\lambda 3}^k/\beta_3]_i & \text{if } i \notin \Omega.\end{cases}$$

In Table 1, we list the correction steps of ADMM-GB, ADMM-Forward, and ADMM-Backward, where they take the constant α as the stepsize. From this table, we can see their difference in computation demand. In particular, for ADMM-GB, the update of x_2^{k+1} is computationally expensive when solving the model (91). More specifically, by denoting $r^k := (A_2^T H A_2)^{-1}A_2^T A_3(x_3^k - \tilde{x}_3^k)$ and recalling that $x_3 = (\mathbf{u}, \mathbf{v}, \mathbf{z})$, we have

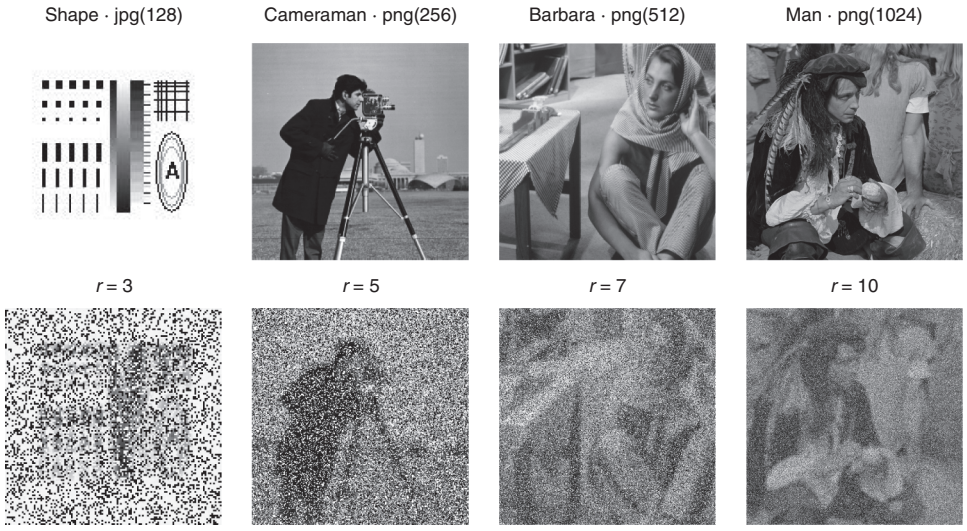
$$\begin{aligned}r^k &:= (A_2^T H A_2)^{-1}A_2^T A_3(x_3^k - \tilde{x}_3^k)\\ &:= (\beta_2^2 I + \beta_3^2 B^T B)^{-1}[\mathbf{y}^k - \tilde{\mathbf{y}}^k - B^T(\mathbf{z}^k - \tilde{\mathbf{z}}^k)]\\ &:= \mathcal{F}^{-1}\left[\frac{\mathcal{F}(\mathbf{y}^k - \tilde{\mathbf{y}}^k) - D^*\circ\mathcal{F}(\mathbf{z}^k - \tilde{\mathbf{z}}^k)}{\beta_2^2 + \beta_3^2|D|^2}\right],\end{aligned}$$

where $\mathcal{F}$ is FFT and $\mathcal{F}^{-1}$ is its inverse; D includes the eigenvalues of the convolutional matrix B and D^* is its conjugate transform, and "$\circ$" is the multiplication in componentwise. Thus the auxiliary matrix D and the constant

Table 1. Correction steps of ADMM-GB, ADMM-Forward, and ADMM-Backward with a constant stepsize α.

Algorithm	Correction step
ADMM-GB	$\begin{cases} x_2^{k+1} = x_2^k + \alpha(\tilde{x}_2^k - x_2^k) - \alpha(A_2^T A_2)^{-1}A_2^T A_3(\tilde{x}_3^k - x_3^k)\\ x_3^{k+1} = x_3^k + \alpha(\tilde{x}_3^k - x_3^k)\\ \lambda^{k+1} = \lambda^k + \alpha(\tilde{\lambda}^k - \lambda^k)\end{cases}$
ADMM-Forward	$\begin{cases} u_2^{k+1} = u_2^k - \alpha(u_2^k - \tilde{u}_2^k)\\ u_3^{k+1} = u_3^k - \alpha(u_2^k - \tilde{u}_2^k + u_3^k - \tilde{u}_3^k)\\ u_\lambda^{k+1} = u_\lambda^k - \alpha(u_\lambda^k - \tilde{u}_\lambda^k - u_2^k + \tilde{u}_2^k - u_3^k + \tilde{u}_3^k)\end{cases}$
ADMM-Backward	$\begin{cases} u_2^{k+1} = u_2^k - \alpha(u_2^k - \tilde{u}_2^k - u_3^k + \tilde{u}_3^k)\\ u_3^{k+1} = u_3^k - \alpha(u_3^k - \tilde{u}_3^k)\\ u_\lambda^{k+1} = u_\lambda^k - \alpha(u_\lambda^k - \tilde{u}_\lambda^k - u_2^k + \tilde{u}_2^k - u_3^k + \tilde{u}_3^k)\end{cases}$

Figure 1. Top row: Clean images. Bottom row: Degraded images by out-of-focus blur with radius r and mixed noise.



$\beta_2^2 + \beta_3^2|D|^2$ can be computed once and then used for all iterations; but two FFT and one inverse FFT must be executed iteratively. This is also the main reason why ADMM-GB performs less efficiently than ADMM-Forward and ADMM-Backward, as the numerical results to be reported show.

For numerical comparisons, we test some images in different sizes varying from 128 to 1,024. All clean images are corrupted by the out-of-focus blur with a radius r. The blurred images are further corrupted by both the impulsive noise with an intensity 0.5 and the zero mean Gaussian noise with a variance 0.01. The clean and degraded images are shown in Figure 1. As in Huang et al. [25], we first apply adaptive median filter (AMF) (see Hwang and Haddad [26]) to identify the set Ω and then remove the impulsive noise within that set. The window size for AMF is taken as 19. The parameters in (90) are fixed as $\tau = 0.003$ and $\rho = 1$ for the model (90).

For the scalars β_i's in (89), we set them as $\beta_1 = \beta_2 = 0.1$ and $\beta_3 = 1$ for all the methods to be tested. For the relaxation parameter γ appearing in (37b) and (49b), we set it as 1.5. In addition, to test the performance of ADMM-Forward and ADMM-Backward with the constant stepsize α in their correction steps, we take $\alpha = 0.5$ as an illustrative example. All the methods take the zero vector as the initial iterate.

To measure the quality of a restored image, we use the signal-to-noise ratio (SNR) in the unit of dB defined as

$$\mathrm{SNR} = 20\log_{10}\frac{\|\mathbf{x}^*\|}{\|\bar{\mathbf{x}} - \mathbf{x}^*\|}, \tag{94}$$

where $\bar{\mathbf{x}}$ is the restored image and $\mathbf{x}^*$ is the clean image. As we have shown in the theoretical analysis (see Remark 4.3), it is reasonable to use $\|u^k - \tilde{u}^k\|^2$ to measure the accuracy of an iterate to a solution point. Thus the stopping criterion for all the methods to be tested is taken as

$$\mathrm{TOL} := \frac{\|u^k - \tilde{u}^k\|^2}{\|u^k\|^2 + 1} < 10^{-4}. \tag{95}$$

In Table 2, we report the numbers of iterations ("It") and computing time in seconds ("CPU") when the methods under test are terminated by satisfying the criterion (95). In addition to the case with chosen stepsizes, the case

Table 2. Numerical results of the image restoration model (90).

	Shape			Cameraman			Barbara			Man		
Algorithm	It	CPU	SNR	It	CPU	SNR	It	CPU	SNR	It	CPU	SNR
EADMM	55	0.61	16.39	44	2.51	17.38	44	11.78	17.49	45	48.98	16.97
ADMM-GB(0.5)	76	1.44 (0.51)	16.38	72	5.62 (2.50)	17.39	60	19.05 (8.19)	17.47	64	83.01 (36.09)	16.99
ADMM-Forward	66	0.76 (0.11)	16.38	42	2.45 (0.66)	17.38	42	10.95 (3.03)	17.50	37	42.03 (10.33)	16.89
ADMM-Forward(0.5)	69	0.69 (0.06)	16.37	46	2.51 (0.48)	17.39	45	11.03 (1.79)	17.50	40	41.53 (5.99)	16.88
ADMM-Backward	50	0.56 (0.06)	16.36	37	1.59 (0.39)	17.32	30	7.91 (2.64)	17.50	30	31.34 (7.34)	16.87
ADMM-Backward(0.5)	67	0.80 (0.08)	16.38	41	1.98 (0.41)	17.12	39	9.86 (1.44)	17.47	38	39.31 (4.66)	16.68

where all the methods take 0.5 as the constant stepsize is reported. The SNR values of the images restored by these methods when the criterion (95) is satisfied are also reported. Since the methods under test mainly differ in their correction steps, we also report the computing time required by their correction steps (accumulated for all iterations), see the data in parenthesis of the CPU columns in Table 2. According to this table, we see that ADMM-Forward and ADMM-Backward are more efficient than ADMM-GB for the tested model—they can achieve solutions in the same level of quality (only having a difference at the first or second digit) measured by the SNR values within shorter time. As we have mentioned, this is mainly because some Fourier transforms are inevitable in the correction steps of ADMM-GB. For the comparison between ADMM-Forward and ADMM-Backward, it seems that ADMM-Backward is slightly faster than ADMM-Forward for most of the scenarios we tested. Moreover, for the tested scenarios, it seems that the constant stepsize 0.5 sometimes can accelerate ADMM-Forward for the tested scenarios. But, in general, it is not conclusive that a constant stepsize must be better than a chosen stepsize—using a constant stepsize saves computation, while the chosen stepsize requiring extra computation can accelerate the contraction to the solution set.

To see the comparison among different methods clearly, we also plot the evolutions of the SNR values of the restored images with respect to iterations and computing times in Figure 2, respectively. In Figure 3, for ADMM-Forward and ADMM-Backward, we plot the evolutions of the SNR values with respect to iterations when different constant stepsizes are chosen. For succinctness, only the Cameraman and Barbara images are tested. Finally, we display the images restored by ADMM-Forward after 50 iterations in Figure 4.

7.2. Example 2—An Image Decomposition Model

Image decomposition is an important problem in image processing and it plays a significant role in many realms such as object recognition and biomedical engineering. A useful image decomposition problem is to decompose a target image into two meaningful components: One is its geometrical part or sketchy approximation, which is called *cartoon* component, and the other is its oscillating part or small-scale special patterns, which is called *texture* component. Mathematically, the cartoon component can be described by a piecewise smooth (or a piecewise constant) function and the texture component is commonly oscillating. Because of their different properties, it is usually required to separate them for further tasks in image processing or image analysis.

We test the model in Ng et al. [34] for decomposing an image with corruptions (e.g., blurry and/or missing pixels):

$$\min_{\mathbf{x}\in\Re^n,\,\mathbf{g}\in\Re^n\times\Re^n}\left\{\tau\||\nabla\mathbf{x}|\|_1+\tfrac{1}{2}\|K(\mathbf{x}+\operatorname{div}\mathbf{g})-\mathbf{f}\|^2+\mu\||\mathbf{g}|\|_\infty\right\},\tag{96}$$

where $\|\cdot\|_1$, $\|\cdot\|$ and $\|\cdot\|_\infty$ denote the l_1, l_2, and l_∞ norms, respectively; $\mathbf{f}\in\Re^n$ is a target image; $\operatorname{div}:=-\nabla^T$ denotes the divergence operator where ∇ is the gradient operator as mentioned in (90); K: $\Re^n\to\Re R^n$ is a linear operator; $\tau>0$ and $\mu>0$ are trade-off constants to balance the target image $\mathbf{f}$ into the cartoon $\mathbf{x}$ and the texture $\mathbf{v}=\operatorname{div}\mathbf{g}$, respectively. Different choices of K correspond to different corruptions in the observed image. For instance, $K=S$, where S is a binary matrix (also the so-called "mask" operator) means decomposing an image with missing pixels, see Ng et al. [34] for more details. In Ng et al. [34], ADMM-GB was used to solve the model (96).

First, we show that the model (96) can be formulated as a special case of (1). In fact, by introducing the auxiliary variables $\mathbf{u}$, $\mathbf{y}$, and $\mathbf{z}$, the model (96) can be rewritten as

$$\begin{array}{ll}\min & \tau\||\mathbf{u}|\|_1+\tfrac{1}{2}\|K\mathbf{y}-\mathbf{f}\|_2^2+\mu\||\mathbf{z}|\|_\infty\\ \text{s.t.} & \mathbf{u}=\nabla\mathbf{x}\\ & \mathbf{y}=\mathbf{x}+\operatorname{div}\mathbf{g}\\ & \mathbf{z}=\mathbf{g},\end{array}\tag{97}$$

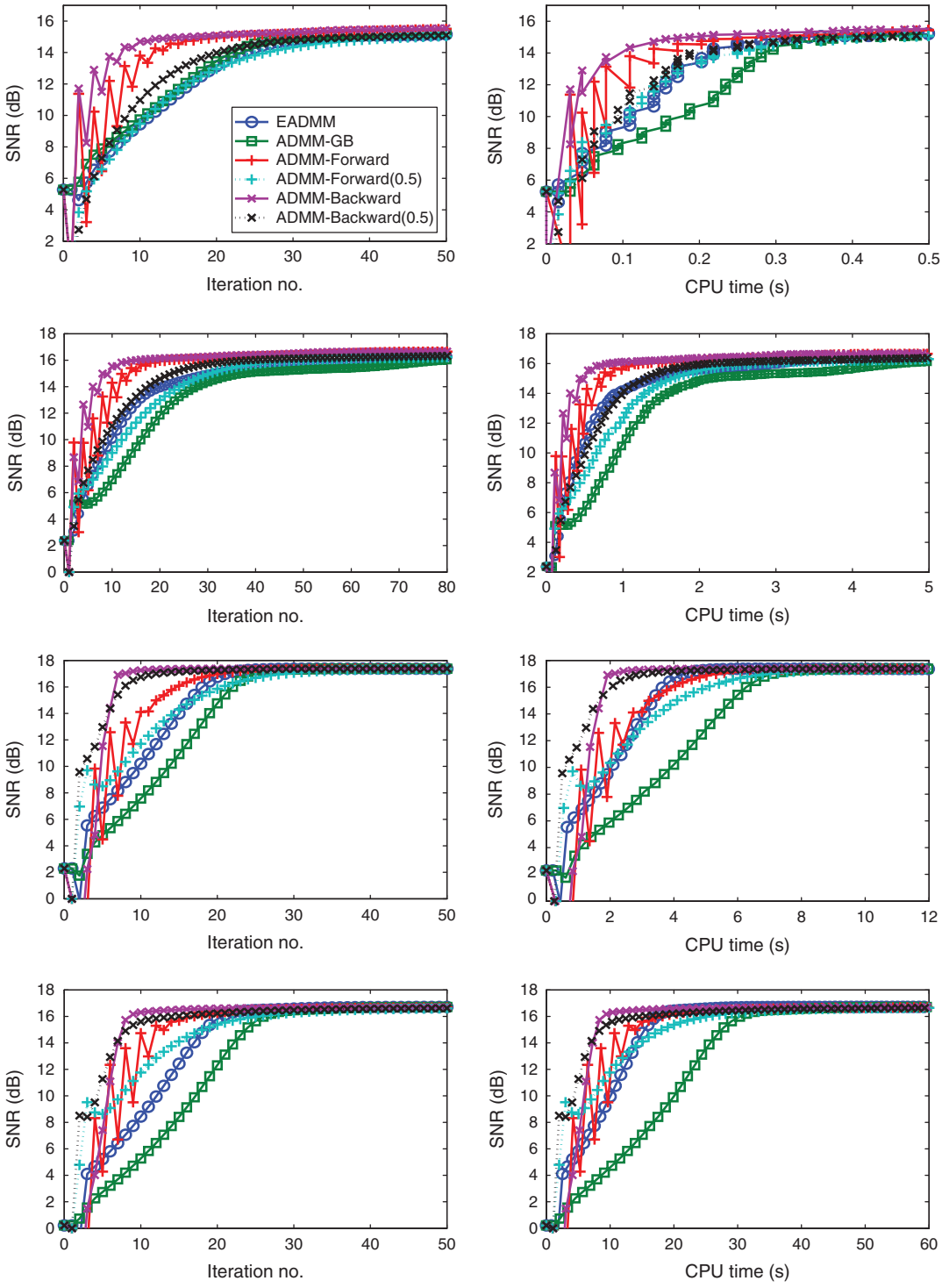
which is a special case of (1) with the following specifications:

- $x_1:=\mathbf{x}$, $x_2:=\mathbf{g}$, $x_3:=(\mathbf{u},\mathbf{y},\mathbf{z})$, $\mathscr{X}_1:=\Re^n$, $\mathscr{X}_2:=\Re^n\times\Re^n$ and $\mathscr{X}_3:=(\Re^n\times\Re^n)\times\Re^n\times(\Re^n\times\Re^n)$;
- $\theta_1(x_1):=0$, $\theta_2(x_2):=0$ and $\theta_3(x_3):=\tau\||\mathbf{u}|\|_1+\tfrac{1}{2}\|K\mathbf{y}-\mathbf{f}\|^2+\mu\||\mathbf{z}|\|_\infty$;
- and

$$A_1:=\begin{bmatrix}\nabla\\ I\\ 0\end{bmatrix},\qquad A_2:=\begin{bmatrix}0\\ \operatorname{div}\\ I\end{bmatrix},\qquad A_3:=\begin{bmatrix}-I & 0 & 0\\ 0 & -I & 0\\ 0 & 0 & -I\end{bmatrix},\qquad b:=0.$$

Then, we elaborate on the minimization subproblems in the ADMM procedure (2) or (20) for solving the model (97).

Figure 2. (Color online) Model (90). Evolutions of SNR w.r.t. to iterations (left column) and computing time (right column). From top to bottom: For Shape, Cameraman, Barbara, and Man images.



- The $\tilde{x}_1$-subproblem, i.e., the $\tilde{\mathbf{x}}^k$-subproblem for (97), corresponds to the following optimization problem:

$$\tilde{x}_1^k = \arg\min_{x_1}\{\|A_1x_1 + H^{1/2}(u_2^k + u_3^k) - u_\lambda^k\|_H^2\},$$
$$\Leftrightarrow \quad (\beta_1^2\nabla^T\nabla + \beta_2^2 I)\tilde{\mathbf{x}}^k = \beta_1\nabla^T[\beta_1 u_{\lambda 1}^k - u_{21}^k - u_{31}^k] + \beta_2(\beta_2 u_{\lambda 2}^k - u_{22}^k - u_{32}^k),$$

which can be easily solved as that of (93).

Figure 3. (Color online) Model (90). Evolutions of SNR w.r.t. iterations for ADMM-Forward and ADMM-Backward with different constant stepsizes α. Left: For Cameraman. Right: For Barbara.

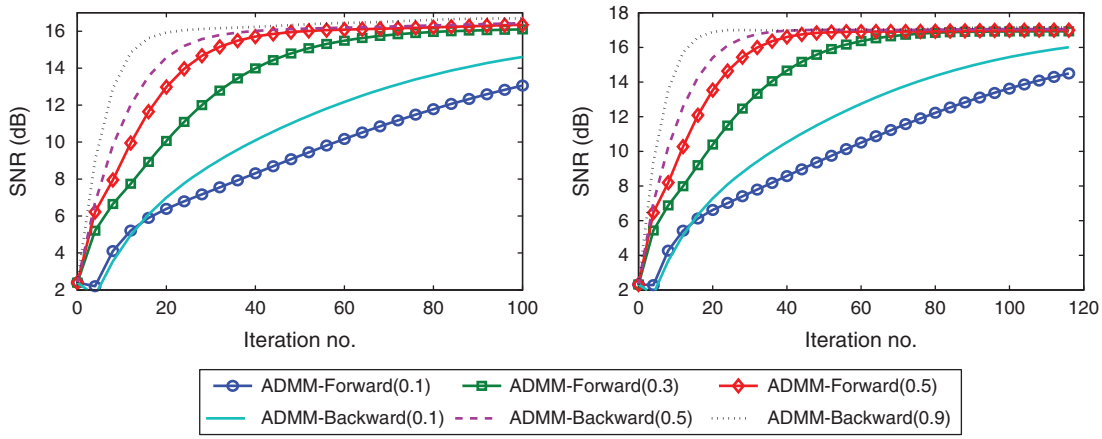


Figure 4. Restored images by ADMM-Forward after 50 iterations for the model (90).

- The $\tilde{x}_2$-subproblem, i.e., the $\tilde{\mathbf{g}}^k$-subproblem for (97), is equivalent to

$$\tilde{x}_2^k = \arg\min_{x_2}\{\|A_1\tilde{x}_1^k + A_2x_2 + H^{-1/2}u_3^k - u_\lambda^k\|_H^2\}$$
$$\Leftrightarrow \quad (\beta_2^2\,\mathrm{div}^T\,\mathrm{div} + \beta_3^2 I)\tilde{\mathbf{g}}^k = \beta_2\,\mathrm{div}^T[\beta_2(u_{\lambda 2}^k - \tilde{x}_1^k) - u_{32}^k] - \beta_3(u_{33}^k - \beta_3 u_{\lambda 3}^k),$$

and the FFT or DCT can be used as that for solving (93) because $\mathrm{div} = -\nabla^T$.

- The $\tilde{x}_3$-subproblem, i.e., the $(\tilde{\mathbf{u}}^k, \tilde{\mathbf{y}}^k, \tilde{\mathbf{z}}^k)$-subproblem for (97), corresponds to

$$\tilde{x}_3^k = (\tilde{\mathbf{u}}^k, \tilde{\mathbf{y}}^k, \tilde{\mathbf{z}}^k) = \arg\min_{\mathbf{u},\mathbf{y},\mathbf{z}}\left\{\tau\||\mathbf{u}|\|_1 + \tfrac{1}{2}\|K\mathbf{y} - \mathbf{f}\|^2 + \mu\||\mathbf{z}|\|_p + \tfrac{1}{2}\|A_1\tilde{\mathbf{x}}^k + A_2\tilde{\mathbf{g}}^k + A_3x_3 - H^{-1/2}u_\lambda^k\|_H^2\right\},$$

and each variable can be solved separably as shown below.

—The $\tilde{\mathbf{u}}$-subproblem can be solved explicitly by the shrinkage operator:

$$\tilde{\mathbf{u}}^k = \arg\min_{\mathbf{u}}\left\{\tau\||\mathbf{u}|\|_1 + \frac{\beta_1^2}{2}\left\|\mathbf{u} - \nabla\tilde{\mathbf{x}}^k + \frac{u_{\lambda 1}^k}{\beta_1}\right\|^2\right\} = \mathrm{shrink}_{\tau/\beta_1^2}\left(\nabla\tilde{\mathbf{x}}^k - \frac{u_{\lambda 1}^k}{\beta_1}\right).$$

—The $\tilde{\mathbf{y}}$-subproblem is equivalent to

$$\tilde{\mathbf{y}}^k = \arg\min_{\mathbf{y}}\left\{\frac{1}{2}\|K\mathbf{y} - \mathbf{f}\|_2^2 + \frac{\beta_2^2}{2}\left\|\tilde{\mathbf{x}}^k + \mathrm{div}\,\tilde{\mathbf{g}}^k - \mathbf{y} - \frac{u_{\lambda 2}^k}{\beta_2}\right\|^2\right\}$$
$$\Leftrightarrow \quad (K^TK + \beta_2^2 I)y = K^T\mathbf{f} + \beta_2[\beta_2(\tilde{\mathbf{x}}^k + \mathrm{div}\,\tilde{\mathbf{g}}^k) - u_{\lambda 2}^k],$$

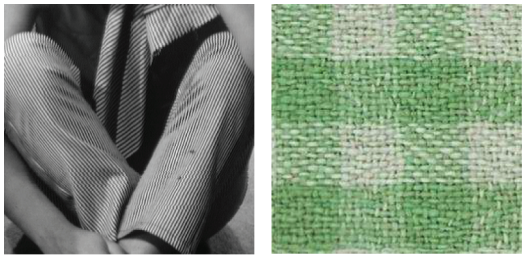
whose computational effort is dependent on the operator K (see Ng et al. [34] for details).

—The $\tilde{\mathbf{z}}$-subproblem is equivalent to

$$\tilde{\mathbf{z}}^k = \arg\min_{\mathbf{z}}\left\{\mu\||\mathbf{z}|\|_\infty + \frac{\beta_3^2}{2}\left\|\mathbf{z} - \tilde{\mathbf{g}}^k - \frac{u_{\lambda 3}^k}{\beta_3}\right\|^2\right\} = \mathrm{prox}_{(\mu/\beta_3^2)\||\cdot|\|_\infty}\left(\tilde{\mathbf{g}}^k + \frac{u_{\lambda 3}^k}{\beta_3}\right), \tag{98}$$

where $\mathrm{prox}_{c\||\cdot|\|_\infty}(\cdot)$ denotes the proximal operator of the function $c\||\cdot|\|_\infty$ for $c > 0$, see, e.g., Martinet [29], Rockafellar [37]. Thus, $\tilde{\mathbf{z}}^k$ is giving by computing the projection onto the l^∞ ball.

Figure 5. (Color online) Test images. Left: 256 × 256 "Barbara.png." Right: 250 × 248 × 3 "weave.jpg."



The correction steps for ADMM-GB, ADMM-Forward, and ADMM-Backward can be summarized similarly as Table 1. For succinctness, we omit them.

For numerical comparison, we test the images with different sizes displayed in Figure 5. For simplification, we only report the cases when $K = I$ in the model (96). The parameters in (96) are taken as $\tau = 0.5$ and $\mu = 5$ in our experiments.

The scalars in (89) are chosen as $\beta_1 = \beta_2 = \beta_3 = 1$ for all the methods to be tested. Again, for the relaxation parameter γ appearing in (37b) and (49b), we set it as 1.5. All the methods take the zero vector as the initial iterate. The stopping criterion for all methods is

$$\text{TOL} = \frac{\|u^k - \tilde{u}^k\|^2}{\|u^k\|^2 + 1} < 10^{-6}. \tag{99}$$

In Table 3, we report the numbers of iteration ("It"), computing time in seconds ("CPU") and the obtained objective function values ("Obj") when the stopping criterion (99) is satisfied for all the methods under test. Note that this example is an image decomposition model; we thus report the objective function values rather than the SNR values as for image restoration models. Again, the data in parenthesis in Table 3 are the accumulative time in seconds of the correction steps for each method. For succinctness, we only report the case where the constant stepsize is chosen as 0.5. Other choices of constant can also be shown similarly as Figure 3; but omitted. Based on the data in Table 3, some conclusions similar as those in the last subsection can be claimed. For example, ADMM-Forward and ADMM-Backward with a chosen or constant stepsize are both more efficient than ADMM-GB, because the latter requires significantly more time for computing the correction steps. For this example, ADMM-Backward is also more efficient than ADMM-Forward for the tested scenarios. In particular, we found that ADMM-Backward is even very competitive with EADMM, which is not necessarily convergent as proved in Chen et al. [6] but usually performs very well numerically.

To see the comparison among these methods clearly, in Figures 6 and 7, we plot the evolutions of the objective function values with respect to iterations and computing time, respectively. For each figure, we zoom in a particular area to display the difference of these methods more clearly. Finally, we display the images recovered by ADMM-Forward after 250 iterations in Figure 8.

8. Conclusions

In this paper, by further studying the combination of the Douglas-Rachford alternating direction method of multipliers (ADMM) with a substitution procedure, we proposed an algorithmic framework for solving a convex minimization model with a general separable structure in its objective function. The proposed forward and backward substitution procedures are computationally inexpensive and thus numerically implementable. Two new algorithms were derived from this algorithmic framework. One of the common features of these two algorithms is

Table 3. Numerical results for the image decomposition model (96).

	Barbara			Weave		
Algorithm	It	CPU	Obj	It	CPU	Obj
EADMM	223	30.95	131.24	190	76.30	352.29
ADMM-GB(0.5)	351	66.91 (29.00)	132.88	334	174.39 (65.94)	360.05
ADMM-Forward	283	53.48 (20.98)	131.36	234	120.71 (45.80)	352.30
ADMM-Forward(0.5)	295	47.33 (16.65)	132.73	246	106.35 (33.85)	358.56
ADMM-Backward	205	32.14 (9.02)	131.02	173	74.69 (16.72)	350.84
ADMM-Backward(0.5)	243	36.46 (10.42)	132.49	209	88.28 (18.80)	357.16

Figure 6. (Color online) Model (96). Evolutions of the objective function value w.r.t. iterations and computing times for Barbara. Right column: Zoom in of the dashed boxes in the left column.

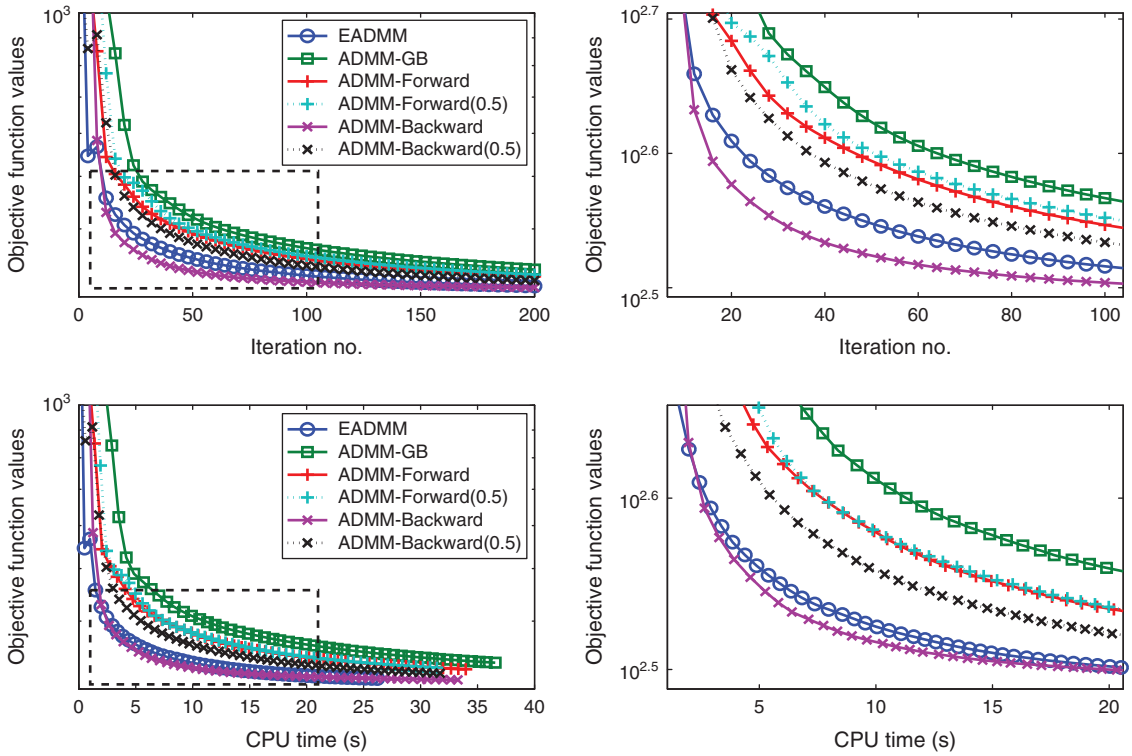


Figure 7. (Color online) Model (96). Evolutions of the objective function value w.r.t. iterations and computing time for weave. Right column: Zoom in of the dashed boxes in the left column.

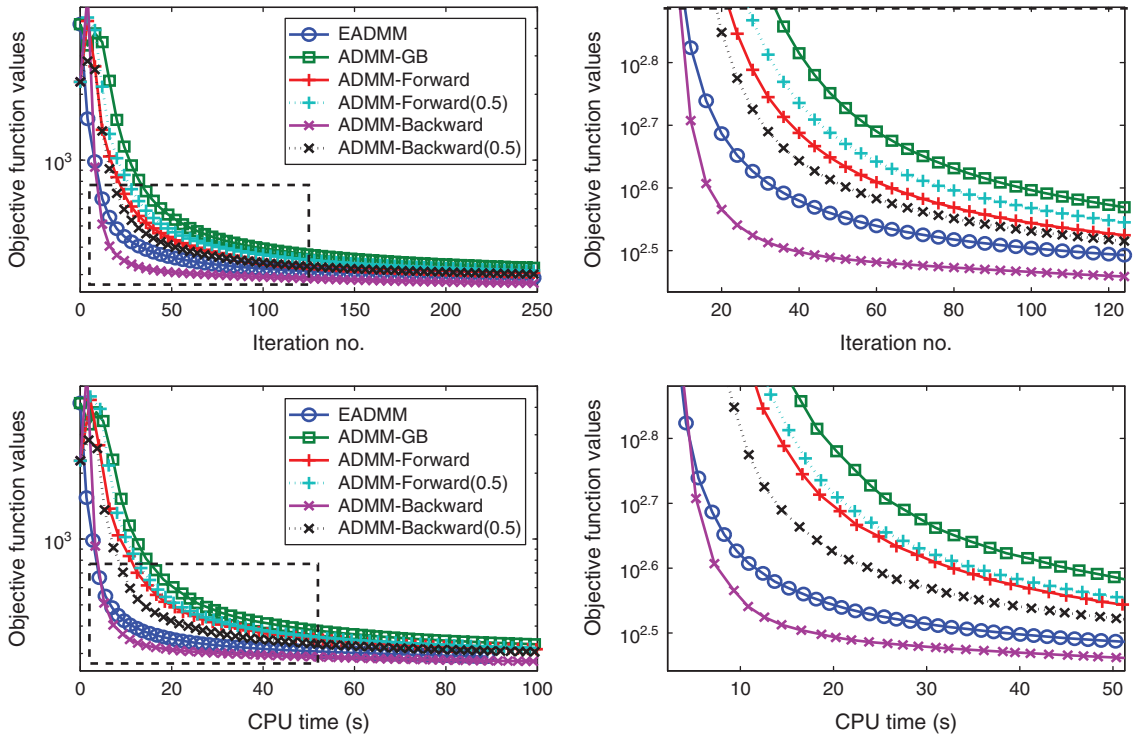
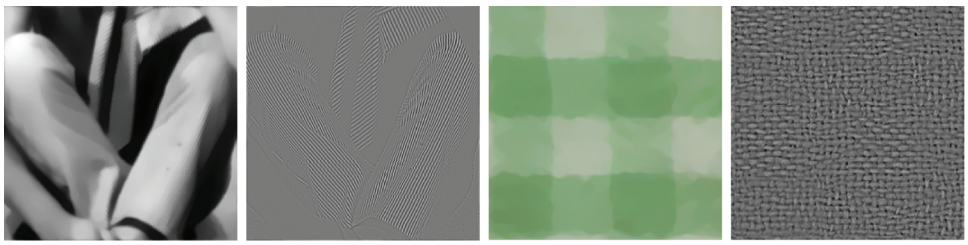


Figure 8. (Color online) Decompositions of cartoons and textures by ADMM-Forward after 250 iterations for model (96). From left to right: Cartoons of Barbara, textures of Barbara, cartoons of weave, and textures of weave.



that they both reduce to the original ADMM for the case of (1) where $m = 2$. For these two algorithms, we proved their global convergence under the analytic framework of contraction methods and derived their local linear convergence rate under some error bound assumptions. Under general settings without additional assumptions, we also estimated some worst-case convergence rates measured by the iteration complexity for the new algorithms. The new algorithms were applied to solve some applications arising in image processing and they were compared with some existing methods of the same kind in the literature. The efficiency of these new algorithms was well demonstrated by the tested examples.

Acknowledgments

The authors thank Dr. Wenxing Zhang for his generous help in completing the numerical experiments.

References

[1] Blum E, Oettli W (1975) *Mathematische Optimierung*, Econometrics and Operations Research, 20 (Springer, Berlin).
[2] Boyd S, Parikh N, Chu E, Peleato B, Eckstein J (2010) Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Machine Learn.* 3(1):1–122.
[3] Chan RH, Yang JF, Yuan XM (2011) Alternating direction method for image inpainting sin wavelet domain. *SIAM J. Image Sci.* 4(3):807–826.
[4] Chan TF, Glowinski R (1978) Finite element approximation and iterative solution of a class of mildly non-linear elliptic equations. Stanford report STAN-CS-78-674, Computer Science Department, Stanford University, Palo Alto, CA.
[5] Chen CH, He BS, Yuan XM (2012) Matrix completion via alternating direction method. *IMA J. Numer. Anal.* 32(1):227–245.
[6] Chen CH, He BS, Ye YY, Yuan XM (2016) The direct extension of ADMM for multi-block convex minimization problems is not necessarily convergent. *Math. Program* 155(1):57–79.
[7] Donoho DL (2006) Compressed sensing. *IEEE Trans. Inform. Theory* 52(4):1289–1306.
[8] Eckstein J, Bertsekas DP (1992) On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Math. Program.* 55(1):293–318.
[9] Esser E (2009) Applications of Lagrangian-based alternating direction methods and connections to split Bregman. UCLA CAM Report, Los Angeles.
[10] Facchinei F, Pang JS (2003) *Finite-Dimensional Variational Inequalities and Complementarity Problems*, Springer Series in Operations Research, Vol. I (Springer, New York).
[11] Fortin M, Glowinski R (1983) Augmented Lagrangian methods: Applications to the numerical solutions of boundary value problems. *Stud. Math. Appl.* Vol. 15 (North-Holland, Amsterdam).
[12] Fukushima M (1992) Application of the alternating direction method of multipliers to separable convex programming problems. *Comput. Optim. Appli.* 1(1):93–111.
[13] Gabay D, Mercier B (1976) A dual algorithm for the solution of nonlinear variational problems via finite-element approximations. *Comput. Math. Appli.* 2(1):17–40.
[14] Glowinski R (1984) *Numerical Methods for Nonlinear Variational Problems* (Springer, New York).
[15] Glowinski R, Le Tallec P (1989) *Augmented Lagrangian and Operator-Splitting Methods in Nonlinear Mechanics*. Studies in Applied Mathematics (SIAM, Philadelphia).
[16] Glowinski R, Marrocco A (1975) Sur l'approximation par éléments finis d'ordre un et résolution par pénalisation-dualité d'une classe de problèmes de Dirichlet non linéaires. *Rev. Fr. Autom. Inf. Rech. Oper. Anal. Numér* R2:41–76.
[17] Han DR, Yuan XM (2012) A note on the alternating direction method of multipliers. *J. Optim. Theory Appli.* 155(1):227–238.
[18] Hansen PC, Nagy JG, O'Leary DP (2006) *Deblurring Images: Matrices, Spectra, and Filtering* (SIAM, Philadelphia).
[19] He BS, Tao M, Yuan XM (2012) Alternating direction method with Gaussian back substitution for separable convex programming. *SIAM J. Optim.* 22(2):313–340.
[20] He BS, Tao M, Yuan XM (2015) A splitting method for separable convex programming. *IMA J. Numer. Anal.* 35(1):394–426.
[21] He BS, Xu MH, Yuan XM (2011) Solving large-scale least squares covariance matrix problems by alternating direction methods. *SIAM J. Matrix Anal. Appli.* 32(1):136–152.
[22] He BS, Liao LZ, Han DR, Yang H (2002) A new inexact alternating directions method for monontone variational inequalities. *Math. Program.* 92(1):103–118.
[23] Hestenes MR (1969) Multiplier and gradient methods. *J. Optim. Theory Appli.* 4(5):303–320.
[24] Hong MY, Luo ZQ (2016) On the linear convergence of the alternating direction method of multipliers. *Math. Programm.* ePub ahead of print July 6, http://doi.org/10.1007/s10107-016-1034-2.

[25] Huang YM, Ng MK, Wen YW (2009) Fast image restoration methods for impulse and Gaussian noise removal. *IEEE Signal Process. Lett.* 16(6):457–460.
[26] Hwang H, Haddad A (1995) Adaptive median filters: New algorithms and results. *IEEE Trans. Image Process.* 4(4):499–502.
[27] Kontogiorgis S, Meyer RR (1998) A variable-penalty alternating directions method for convex optimization. *Math. Program.* 83(1): 29–53.
[28] Lan G, Monteiro RDC (2016) Iteration-complexity of first-order augmented Lagrangian methods for convex programming. *Math. Program.* 155(1):511–547.
[29] Martinet B (1970) Regularision d'inéquations variationnelles par approximations successive. *Revue Francaise d'Automatique et Informatique Recherche Opérationnelle* 126:154–159.
[30] Necoara I, Suykens J (2008) Application of a smoothing technique to decomposition in convex optimization. *IEEE Trans. Auto. Control* 53(11):2674–2679.
[31] Nedelcu V, Necorara I, Tran-Ding Q (2014) Computational complexity of inexact gradient augmented Lagrangian methods: Application to constrained MPC. *SIAM J. Control Optim.* 52(5):3109–3134.
[32] Ng MK, Wang F, Yuan XM (2011) Inexact alternating direction methods for image recovery. *SIAM J. Sci. Comput.* 33(4):1643–1668.
[33] Ng MK, Weiss PA, Yuan XM (2010) Solving constrained total-variation problems via alternating direction methods. *SIAM J. Sci. Comput.* 32(5):2710–2736.
[34] Ng MK, Yuan XM, Zhang WX (2013) On variational image decomposition model for blurred images with missing pixel values. *IEEE Trans. Image Processing* 22(6):2233–2246.
[35] Peng YG, Ganesh A, Wright J, Xu WL, Ma Y (2012) Robust alignment by sparse and low-rank decomposition for linearly correlated images. *IEEE Tran. Pattern Anal. Machine Intelligence* 34(11):2233–2246.
[36] Powell MJD (1969) A method for nonlinear constraints in minimization problems. Fletcher R, ed. *Optimization* (Academic Press, New York), 283–298.
[37] Rockafellar RT (1976) Augmented Lagrangians and applications of the proximal point algorithm in convex programming. *Math. Oper. Res.* 1(2):97–116.
[38] Rudin L, Osher S, Fatemi E (1992) Nonlinear total variation based noise removal algorithms. *Physica D.* 60(1-4):259–268.
[39] Ruszczyński A (1993) Parallel decomposition of multistage stochastic programming problems. *Math. Program.* 58(1):201–228.
[40] Sun J, Zhang S (2010) A modified alternating direction method for convex quadratically constrained quadratic semidefinite programs. *Eur. J. Oper. Res.* 207(3):1210–1220.
[41] Tao M, Yuan XM (2011) Recovering low-rank and sparse components of matrices from incomplete and noisy observations. *SIAM J. Optim.* 21(1):57–81.
[42] Tibshirani R, Saunders M, Rosset S, Zhu J, Knight K (2005) Sparsity and smoothness via the fused lasso. *J. Royal Statist. Soc.* 67(1):91–108.
[43] Tran-Dinh Q, Necoara I, Diehl M (2014) Path-following gradient-based decomposition algorithms for separable convex optimization. *J. Global Optim.* 59(1):59–80.
[44] Tran-Dinh Q, Necoara I, Savorgnans C, Diehl M (2013) An inexact perturbed path-following method for Lagrangian decomposition in large-scale separable convex optimization. *SIAM J. Optim.* 23(1):95–125.
[45] Tseng P (1991) Applications of a splitting algorithm to decomposition in convex programming and variational inequalities. *SIAM J. Con. Optim.* 29(1):119–138.
[46] Zhang S, Ang J, Sun J(2013) An alternating direction method for solving convex nonlinear semidefinite programming problems. *Optim.* 62(4):527–543.
[47] Zhang XQ, Burger M, Osher S (2010) A unified primal-dual algorithm framework based on Bregman iteration. *J. Sci. Comput.* 46(1):20–46.
[48] Zhou Z, Li X, Wright J, Candes EJ, Ma Y (2010) Stable principal component pursuit. *Proc. IEEE Intern. Sympos. Inform. Theory, ISIT '10* (IEEE, Piscataway, NJ) 1518–1522.