A splitting method for separate convex programming with linking linear constraints

Bingsheng He[1] **Min Tao**[2] and **Xiaoming Yuan**[3]

June 25, 2010

Abstract. We consider the separate convex programming problem with linking linear constraints, where the objective function is in the form of the sum of m individual functions without crossed variables. The special case with $m = 2$ has been well studied in the literature and some algorithms are very influential, e.g. the alternating direction method. The research for the case with general m, however, is in completely infancy, and only a few methods are available in the literature. To generate a new iterate, almost all existing methods produce temporary iterates via solving m subproblems generated by splitting the corresponding augmented Lagrangian function in either parallel or alternating way, and then apply some correction steps to correct the temporary iterates. These correction steps, however, may result in critical difficulties for some applications of the model under consideration. In this paper, we develop the first method requiring no correction steps at all to generate new iterates for the model with general m. We also apply the new method to solve some concrete applications arising in image processing and Statistics, and report the promising numerical performance.

Keywords: Convex programming, separate structure, linking constraint, augmented Lagrangian method, total variation, image restoration

---

1 Introduction

We consider the separate convex programming problem with linking linear constraints, where the objective function is in the form of the sum of m individual functions without crossed variables:

$$\min\left\{\sum_{i=1}^{m}\theta_i(x_i)\ \middle|\ \sum_{i=1}^{m}A_ix_i=b;\ x_i\in\mathcal{X}_i,\ i=1,2,\ldots,m\right\}, \tag{1.1}$$

where $\theta_i : \Re^{n_i} \rightarrow \Re$ $(i = 1, 2, \cdots, m)$ are closed proper convex functions (not necessarily smooth); $A_i \in \Re^{l\times n_i}$ $(i = 1, 2, \cdots, m)$; $\mathcal{X}_i \subseteq \Re^{n_i}$ $(i = 1, 2, \cdots, m)$ are closed convex sets; $b \in \Re^l$ and $\sum_{i=1}^m n_i = n$. Throughout, we assume that the solution set of (1.1) in nonempty.

In the literature the special case of (1.1) with $m = 2$ has been well studied, and one of the most popular methods is the alternating direction method (ADM) contributed originally in [19, 20] and relevant closely to the Peaceman-Rachford and Douglas-Rachford operator splitting method [11, 44]. More specifically, to solve

$$\min\{\theta_1(x_1) + \theta_2(x_2) \mid A_1x_1 + A_2x_2 = b,\ x_i \in \mathcal{X}_i,\ i = 1, 2\}, \tag{1.2}$$

[1]Department of Mathematics, Nanjing University, China and National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China. This author was supported by the NSFC Grant 10971095, the NSF of Province Jiangsu grant BK2008255 and the Cultivation Fund of the Key Scientific and Technical Innovation Project, Ministry of Education of China (No.708044) . Email: hebma@nju.edu.cn
[2]Department of Mathematics, Nanjing University, Nanjing, 210093, China.
[3]Corresponding author. Department of Mathematics, Hong Kong Baptist University, Hong Kong, China. This author was supported in part by an HK RGC grant. Email: xmyuan@hkbu.edu.hk

the ADM splits the corresponding augmented Lagrangian function of (1.2) in the alternating manner such that the variables x_1 and x_2 can be minimized separately in the alternating order. The iterative scheme of ADM for (1.2) is:

$$\begin{cases} x_1^{k+1} = \operatorname{Argmin}\left\{\theta_1(x_1) - (\lambda^k)^T(A_1x_1 + A_2x_2^k - b) + \frac{\beta}{2}\|A_1x_1 + A_2x_2^k - b\|^2 \mid x \in \mathcal{X}_1\right\}; \\ x_2^{k+1} = \operatorname{Argmin}\left\{\theta_2(x_2) - (\lambda^k)^T(A_1x_1^{k+1} + A_2x_2 - b) + \frac{\beta}{2}\|A_1x_1^{k+1} + A_2x_2 - b\|^2 \mid x_2 \in \mathcal{X}_2\right\}; \\ \lambda^{k+1} = \lambda^k - \beta(A_1x_1^{k+1} + A_2x_2^{k+1} - b). \end{cases}$$

where λ^k is the Lagrangian multiplier associated with the linear constraints and $\beta > 0$ the penalty parameter for the violation of the linear constraints. We refer to, e.g. [2, 9, 12, 13, 16, 18, 21, 22, 28, 33, 49], for the intensive study of ADM in the literature of convex programming and variational inequalities (VIs). In particular, as analyzed in [15], the ADM has inspired the influential linearized Bregman method in [25] which is very popular in the area of image processing. Very recently, some novel applications of ADM have been discovered by various authors, e.g. the total-variation problem in image processing [1, 15, 38, 52], the covariance selection problem and semidefinite least square problem [30, 54], the semidefinite programming problems [47, 50], and the sparse and low-rank recovery problem [35, 55].

Another splitting strategy for (1.2) is to decompose the corresponding augmented Lagrangian function in the parallel manner such that the individual variables $x_i's$ are eligible for simultaneous computation if advanced parallel computing infrastructure is available. The basic iterative scheme of parallel splitting methods for (1.2) is

$$\begin{cases} x_1^{k+1} = \operatorname{Argmin}\left\{\theta_1(x_1) - (\lambda^k)^T(A_1x_1 + A_2x_2^k - b) + \frac{\beta}{2}\|A_1x_1 + A_2x_2^k - b\|^2 \mid x_1 \in \mathcal{X}_1\right\}, \\ x_2^{k+1} = \operatorname{Argmin}\left\{\theta_2(x_2) - (\lambda^k)^T(A_1x_1^k + A_2x_2 - b) + \frac{\beta}{2}\|A_1x_1^k + A_2x_2 - b\|^2 \mid x_2 \in \mathcal{X}_2\right\}, \\ \lambda^{k+1} = \lambda^k - \beta(A_1x_1^{k+1} + A_2x_2^{k+1} - b), \end{cases}$$

where the variables $x_i's$ are not coupled and thus can be solved simultaneously. We refer to, e.g. [7, 9, 23, 27, 34], for the parallel splitting methods for solving (1.2).

The more general case of (1.1) where $m \geq 3$ captures many concrete applications. To mention a few, the problem of recovering the low-rank and sparse components of matrices from incomplete and noisy observation launched in [48], the constrained total-variation image restoration and reconstruction problem in [38, 43] and the minimal surface PDE problem in [51] can be easily reformulated into (1.1) with $m = 3$; the constrained total-variation superresolution image reconstruction problem [5, 37] is in the form of (1.1) with $m = 4$; the multistage stochastic programming problem in [42] is modeled into (1.1) with $m = 6$; and the deblurring Poissonian image processing problem [45] is characterized by (1.1) with general m.

These attractive applications urge to develop efficient numerical algorithms for solving the general case of (1.1) with $m \geq 3$. For this purpose, inspired by the efficiency of ADM, the instant idea is to split the augmented Lagrangian function of (1.1) exactly in the alternating manner such that the variables $x_i's$ can be minimized separately in the alternative order. This yields the following

ADM-like iterative scheme:

$$\left\{\begin{array}{l} x_1^{k+1} = \text{Argmin}\Big\{\theta_1(x_1) - (\lambda^k)^T p_1(x_1) + \frac{\beta}{2}\|p_1(x_1)\|^2 \;\Big|\; x_1 \in \mathcal{X}_1\Big\}; \\ \cdots\cdots \\ x_i^{k+1} = \text{Argmin}\Big\{\theta_i(x_i) - (\lambda^k)^T p_i(x_i) + \frac{\beta}{2}\|p_i(x_i)\|^2 \;\Big|\; x_i \in \mathcal{X}_i\Big\}; \\ \cdots\cdots \\ x_m^{k+1} = \text{Argmin}\Big\{\theta_m(x_m) - (\lambda^k)^T p_m(x_m) + \frac{\beta}{2}\|p_m(x_m)\|^2 \;\Big|\; x_m \in \mathcal{X}_m\Big\}; \\ \lambda^{k+1} = \lambda^k - \beta(\sum_{j=1}^m A_j x_j^{k+1} - b), \end{array}\right. \tag{1.3}$$

where

$$p_i(x_i) = \sum_{j=1}^{i-1} A_j x_j^{k+1} + A_i x_i + \sum_{j=i+1}^{m} A_j x_j^k - b, \quad i = 1, \ldots, m,$$

Unfortunately, the validity of the ADM extension (1.3) is so far not clear. In fact, even for the special case of (1.3) where $m = 3$, the convergence of the extended ADM (1.3) is still ambiguous. In [29], the ADM for (1.2) was extended to (1.1) with general m in the sense that the iterate generated by the ADM-like scheme (1.3) served as a temporary iterate and then it was used to construct a descent direction along which the new iterate is yielded. Thus, an ADM-based method for (1.1) was developed in the prediction-correction fashion, where the ADM-like scheme (1.3) produces the predictor and the descent step is the correction step generating the new iterate.

When the parallel splitting strategy is applied to (1.1), the resulted subproblems at each iteration are:

$$\left\{\begin{array}{l} x_1^{k+1} = \text{Argmin}\Big\{\theta_1(x_1) - (\lambda^k)^T q_1(x_1) + \frac{\beta}{2}\|q_1(x_1)\|^2 \;\Big|\; x_1 \in \mathcal{X}_1\Big\}; \\ \cdots\cdots \\ x_i^{k+1} = \text{Argmin}\Big\{\theta_i(x_i) - (\lambda^k)^T q_i(x_i) + \frac{\beta}{2}\|q_i(x_i)\|^2 \;\Big|\; x_i \in \mathcal{X}_i\Big\}; \\ \cdots\cdots \\ x_m^{k+1} = \text{Argmin}\Big\{\theta_m(x_m) - (\lambda^k)^T q_m(x_m) + \frac{\beta}{2}\|q_m(x_m)\|^2 \;\Big|\; x_m \in \mathcal{X}_m\Big\}; \\ \lambda^{k+1} = \lambda^k - \beta(\sum_{j=1}^m A_j x_j^{k+1} - b), \end{array}\right. \tag{1.4}$$

where

$$q_i(x_i) = \sum_{j=1}^{i-1} A_j x_j^k + A_i x_i + \sum_{j=i+1}^{m} A_j x_j^k - b. \quad i = 1, \ldots, m.$$

We refer to [27, 48] for some parallel methods solving (1.1) with $m = 3$. In [26], a parallel splitting method for (1.1) with general m was developed also in the prediction-correction fashion, where the iterate generated by (1.4) was used as a predictor and a descent step is employed as the correction step to generate a new iterate.

Therefore, almost all the existing methods for solving the general case of (1.1) with $m \geq 3$ cannot use directly the iterate generated by (1.3) or (1.4) as the new iterate, and they all require correction steps to correct the iterate generated by (1.3) or (1.4). To the best of our knowledge, the only exception is our variant alternating splitting augmented Lagrangian method (VASALM) in [48] for the particular case of (1.1) with $m = 3$ arising in the sparse and low-rank matrix recovery problem (see [6, 8]). As we analyzed for the sparse and low-rank recovery problem with incomplete and noisy observation in [48] (see the PSALM therein), these correction steps may cause critical difficulties for remaining the low-rank property for recovered low-rank components obtained by (1.4). More specifically, even if the recovered low-rank components obtained by (1.4) are of low rank, we usually end up with high-rank new iterates after implementing some correction steps. We believe that the reason causing this difficulty is that the additional correction steps usually destroy the low-rank feature of the temporary iterate generated by (1.4). In Section 5.1, we shall implement the method in [29] to solve the the sparse and low-rank recovery problem with incomplete and noisy observation in [48], and illustrate that correction steps are also very likely to destroy the low-rank feature of the temporary iterate generated by (1.3).

These difficulties arising in applications of (1.1) motive us to develop such a splitting method without correction steps for solving the most general case of (1.1) with $m \geq 3$, and this is the main motivation of this paper. More specifically, by embedding both the alternating and the parallel strategies in the way of splitting of the augmented Lagrangian function of (1.1), we develop a novel splitting method for solving (1.1) without any correction step. At each iteration, the new splitting method solves the variables $x_i's$ separately via solving m smaller and easier subproblems individually, among which the variables $x_i's$ $(i = 2, \cdots, m)$ are eligible for simultaneous computation once x_1 is computed and the Lagrangian multiplier is updated accordingly. Another fact urging splitting methods without correction steps is that some concrete applications of (1.1) are often large-scale in dimensions, and thus the required correction steps may result in additional computation considerately. We will illustrate it in Section 5.2.

Finally, we mention some interesting methods for solving some problems which are revelent to but essentially different from (1.1). For example, the alternating linearization approach in [23, 32] for minimizing the sum of two convex functions without constraints and [24] for minimizing the sum of finitely many convex functions without constraints; some splitting methods for signal processing problems which can be modeled as minimizing the sum of finitely many convex functions without constraints in [10]; the projective splitting method in [14] and the Jacobian-like method in [46] for finding zero points of the sum of finitely many maximal monotone operators without constraints.

The rest of the paper is organized as follows. In Section 2, we provide some preliminary useful for the convergence analysis. In Section 3, we present the new method followed by some remarks. We then analyze the convergence of the new method in Section 4. In Section 5, we apply the new method to solve some application problems arising in image processing and Statistics. Finally, some conclusions are made in Section 6.

2 Preliminaries

In this section, we summarize some basic properties and related definitions that will be used in the coming analysis and discussions.

A mapping $F : \mathcal{X} \subset \Re^n \to \Re^n$ is said to be monotone respect to $\mathcal{X}$ if

$$(x - y)^T (F(x) - F(y)) \geq 0, \quad \forall\, x, y \in \mathcal{X}.$$

For $i = 1, 2, \cdots, m$, let $f_i(x_i) \in \partial(\theta_i(x_i))$ where $\partial(\cdot)$ denotes the subgradient operator of a convex function. Then it is well known that f_i is monotone when $\theta_i(x_i)$ is convex.

By deriving the optimal condition, it is easy to see that the separate convex programming with linking linear constraints (1.1) is characterized by the following variational inequality:

$$\text{Find } \; w^* = (x_1^*, x_2^*, ..., x_m^*, \lambda^*) \in \mathcal{W} := \mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_m \times \Re^l$$

such that

$$\begin{cases} (x_1^{'} - x_1^*)^T\{f_1(x_1^*) - A_1^T\lambda^*\} \geq 0, \\ (x_2^{'} - x_2^*)^T\{f_2(x_2^*) - A_2^T\lambda^*\} \geq 0, \\ \vdots \\ (x_m^{'} - x_m^*)^T\{f_m(x_m^*) - A_m^T\lambda^*\} \geq 0, \\ (\lambda^{'} - \lambda^*)^T(\sum_{i=1}^m A_i x_i^* - b) \geq 0, \end{cases} \qquad \forall\, w^{'} = (x_1^{'}, x_2^{'}, \cdots, x_m^{'}, \lambda^{'}) \in \mathcal{W}. \tag{2.1}$$

Or, we can rewrite (2.1) into the following more compact form denoted by VI$(F, \mathcal{W})$:

$$(w' - w^*)^T F(w^*) \geq 0, \quad \forall\, w' \in \mathcal{W}, \tag{2.2}$$

where

$$w = \begin{pmatrix} x \\ \lambda \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \\ \lambda \end{pmatrix}, \quad F(w) = \begin{pmatrix} f_1(x_1) - A_1^T\lambda \\ f_2(x_2) - A_2^T\lambda \\ \vdots \\ f_m(x_m) - A_m^T\lambda \\ \sum_{i=1}^m A_i x_i - b \end{pmatrix}, \tag{2.3}$$

and

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

Note that $F(w)$ defined in (2.3) is monotone whenever $f_i's$ are.

Under the aforementioned assumption that the solution set of (1.1) is nonempty, the solution set of (2.2)-(2.3) denoted by $\mathcal{W}^*$ is also nonempty and convex (see Theorem 2.3.5 in [17]).

In addition, for the monotone VI$(F, \mathcal{W})$, we have

$$(w - w^*)^T F(w) \geq (w - w^*)^T F(w^*), \quad \forall\, w \in \mathcal{W},\ w^* \in \mathcal{W}^*.$$

Consequently, because

$$(w - w^*)^T F(w^*) \geq 0, \quad \forall w \in \mathcal{W},$$

we obtain

$$(w - w^*)^T F(w) \geq 0, \quad \forall\, w \in \mathcal{W}. \tag{2.4}$$

With the notation of $v = (x_2, \cdots, x_m, \lambda)$ and $\mathcal{W}^*$, for convenience, we also define

$$\mathcal{V} = \mathcal{X}_2 \times \cdots \times \mathcal{X}_m \times \Re^l, \qquad \Lambda^* = \{\lambda^* \,|\, (x_1^*, x_2^*, \ldots, x_m^*, \lambda^*) \in \mathcal{W}^*\},$$

and

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

3 The new method

In this section, we present the new splitting method for solving (1.1) and give some remarks.

Let $H \in \Re^{l\times l}$ be a positive definite matrix. In particular, we can take $H = \beta I_l$ where $\beta > 0$ is a constant and I_l denotes the identity matrix in $\Re^{l\times l}$. With the given $v^k = (x_2^k, \cdots, x_m^k, \lambda^k)$, the new splitting method for (1.1) generates the new iterative $(x_1^{k+1}, x_2^{k+1}, \cdots, x_m^{k+1}, \lambda^{k+1})$ via the following scheme.

Algorithm 1: the new splitting method for solving (1.1) with $m \geq 3$:

Step 1. Solve x_1^{k+1} via:

$$\min\{\theta_1(x_1) - (\lambda^k)^T A_1 x_1 + \frac{1}{2}\|A_1 x_1 + \sum_{i=2}^{m} A_i x_i^k - b\|_H^2 \mid x_1 \in \mathcal{X}_1\}, \tag{3.1}$$

Step 2. Update the Lagrangian multiplier with x_1^{k+1}:

$$\lambda^{k+\frac{1}{2}} = \lambda^k - H\big(A_1 x_1^{k+1} + \textstyle\sum_{i=2}^{m} A_i x_i^k - b\big). \tag{3.2}$$

Step 3. Solve x_i^{k+1} $(i = 2, \cdots, m)$ via (if possible, simultaneously):

$$\min\{\theta_i(x_i) - (\lambda^{k+\frac{1}{2}})^T A_i x_i + \frac{\mu}{2}\|A_i(x_i - x_i^k)\|_H^2 \mid x_i \in \mathcal{X}_i\}, \tag{3.3}$$

where the parameter μ satisfies the **Assumption** specified below.

Step 4. Update the Lagrangian multiplier with x_i^{k+1} $(i = 1, 2, \cdots, m)$:

$$\lambda^{k+1} = \lambda^k - H\big(\textstyle\sum_{i=1}^{m} A_i x_i^{k+1} - b\big). \tag{3.4}$$

Assumption:

The parameter μ in (3.3) is chosen such that

$$\|v^k - v^{k+1}\|_G^2 + 2(\lambda^k - \lambda^{k+1})^T\big(\sum_{i=2}^{m} A_i(x_i^k - x_i^{k+1})\big) \geq \tau\|v^k - v^{k+1}\|_G^2 \tag{3.5}$$

is satisfied for a constant $\tau > 0$, where

$$v^k = \begin{pmatrix} x_2^k \\ \vdots \\ x_m^k \\ \lambda^k \end{pmatrix} \quad \text{and} \quad G = \begin{pmatrix} \mu A_2^T H A_2 & & & \\ & \ddots & & \\ & & \mu A_m^T H A_m & \\ & & & H^{-1} \end{pmatrix}. \tag{3.6}$$

Note that we abuse the notation of $\|\cdot\|_G$ to denote that

$$\|v^k - v^{k+1}\|_G = \big(\mu\textstyle\sum_{i=2}^{m}\|A_i(x_i^k - x_i^{k+1})\|_H^2 + \|\lambda^k - \lambda^{k+1}\|_{H^{-1}}^2\big)^{1/2},$$

even though that the matrix G defined in (3.6) is only a block diagonal positive semi-definite matrix. This abuse to simplify notation does not invalidate the following analysis.

To illustrate that the proposed method is well defined and easily implementable, we first need to show that the **Assumption** can be easily achieved. In the following, we demonstrate that the **Assumption** is guaranteed to be satisfied whenever μ is simply chosen as:

$$\mu > m - 1.$$

Note that

$$\|v^k - v^{k+1}\|_G^2 + 2(\lambda^k - \lambda^{k+1})^T\big(\sum_{i=2}^{m} A_i(x_i^k - x_i^{k+1})\big)$$

$$= \begin{pmatrix} H^{1/2}A_2(x_2^k - x_2^{k+1}) \\ H^{1/2}A_3(x_3^k - x_3^{k+1}) \\ \vdots \\ H^{1/2}A_m(x_m^k - x_m^{k+1}) \\ H^{-1/2}(\lambda^k - \lambda^{k+1}) \end{pmatrix}^T \begin{pmatrix} \mu I_l & 0 & \cdots & 0 & I_l \\ 0 & \mu I_l & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 & I_l \\ 0 & \cdots & 0 & \mu I_l & I_l \\ I_l & \cdots & I_l & I_l & I_l \end{pmatrix} \begin{pmatrix} H^{1/2}A_2(x_2^k - x_2^{k+1}) \\ H^{1/2}A_3(x_3^k - x_3^{k+1}) \\ \vdots \\ H^{1/2}A_m(x_m^k - x_m^{k+1}) \\ H^{-1/2}(\lambda^k - \lambda^{k+1}) \end{pmatrix} \quad (3.7)$$

and

$$\|v^k - v^{k+1}\|_G^2 = \begin{pmatrix} H^{1/2}A_2(x_2^k - x_2^{k+1}) \\ H^{1/2}A_3(x_3^k - x_3^{k+1}) \\ \vdots \\ H^{1/2}A_m(x_m^k - x_m^{k+1}) \\ H^{-1/2}(\lambda^k - \lambda^{k+1}) \end{pmatrix}^T \begin{pmatrix} H^{1/2}A_2(x_2^k - x_2^{k+1}) \\ H^{1/2}A_3(x_3^k - x_3^{k+1}) \\ \vdots \\ H^{1/2}A_m(x_m^k - x_m^{k+1}) \\ H^{-1/2}(\lambda^k - \lambda^{k+1}) \end{pmatrix}.$$

Therefore, the assumption is satisfied when the $ml \times ml$ matrix

$$\widetilde{M} = \begin{pmatrix} \mu I_l & 0 & \cdots & 0 & I_l \\ 0 & \mu I_l & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 & I_l \\ 0 & \cdots & 0 & \mu I_l & I_l \\ I_l & \cdots & I_l & I_l & I_l \end{pmatrix}_{ml \times ml}$$

is positive definite.

Note that the matrix $\widetilde{M}$ has the same largest (resp. smallest) eigenvalues as the $m \times m$ symmetric matrix

$$M = \begin{pmatrix} \mu & 0 & \cdots & 0 & 1 \\ 0 & \mu & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 & 1 \\ 0 & \cdots & 0 & \mu & 1 \\ 1 & \cdots & 1 & 1 & 1 \end{pmatrix}_{m \times m}. \quad (3.8)$$

Lemma 3.1 *For $m \geq 2$, the $m \times m$ symmetric matrix M defined in* (3.8) *has $(m-2)$ multiple eigenvalues*

$$\nu_1 = \nu_2 = \cdots = \nu_{m-2} = \mu,$$

and another two eigenvalues

$$\nu_{m-1}, \nu_m = \frac{(\mu+1) \pm \sqrt{(\mu+1)^2 + 4((m-1)-\mu)}}{2}.$$

Proof. Let e be a $(m-1)$-vector whose each element equals 1. Thus

$$M = \begin{pmatrix} \mu I_{m-1} & e \\ e^T & 1 \end{pmatrix}.$$

Without loss of generality, we assume that the eigenvectors of M have forms

$$z = \begin{pmatrix} y \\ 0 \end{pmatrix} \qquad \text{or} \qquad z = \begin{pmatrix} y \\ 1 \end{pmatrix},$$

where $y \in \Re^{m-1}$. In the first case, we have

$$\begin{cases} \mu y = \nu y, \\ e^T y = 0. \end{cases} \tag{3.9}$$

It is clear that the $(m-1)$-vectors

$$y^1 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \\ -1 \end{pmatrix}, \quad y^2 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \\ -1 \end{pmatrix} \qquad \cdots\cdots \qquad y^{m-2} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \\ -1 \end{pmatrix}$$

are linear independent and satisfy (3.9) with $\nu = \mu$. Thus,

$$z^i = \begin{pmatrix} y^i \\ 0 \end{pmatrix}, \quad i = 1, \ldots, m-2,$$

are eigenvectors of M and the related eigenvalue

$$\nu_1 = \nu_2 = \cdots = \nu_{m-2} = \mu.$$

In the second case, $z^T = (y^T, 1)$, we have

$$\begin{cases} \mu y + e = \nu y, \\ e^T y + 1 = \nu. \end{cases} \tag{3.10}$$

It follows from (3.10) that

$$(\nu - \mu)(\nu - 1) - (m-1) = 0,$$

and thus

$$\nu_{m-1}, \nu_m = \frac{(\mu+1) \pm \sqrt{(\mu+1)^2 + 4((m-1) - \mu)}}{2}.$$

The lemma is proved. $\Box$

For $\mu \geq 1$, it is easy to verify that

$$\nu_m = \frac{(\mu+1) - \sqrt{(\mu+1)^2 + 4((m-1) - \mu)}}{2} \tag{3.11}$$

is the smallest eigenvalue of M. In addition, $\nu_m > 0$ if and only if $\mu > (m-1)$. Therefore, $\mu > m-1$ is a sufficient condition to guarantee that (3.5) to be satisfied with $\tau = \nu_m > 0$.

In the end, we give some remarks on the proposed method.

Remark 3.2 *For the case that $m = 1$, the proposed method is exactly the classical augmented Lagrangian method. For the case that $m = 2$, if we take the parameter $\mu = 1$, then ADM is recovered. Also, the VASALM proposed in [48] is a special case of Algorithm 1.*

4 Convergence

In this section, we mainly prove the convergence of the proposed Algorithm 1. Before that, we emphasize that like the existing prediction-correction methods in [26, 29] for (1.1), the proposed Algorithm 1 without correction step can be easily extended to a new prediction-correction method where the iterate generated by Algorithm 1 is used as a predictor which is corrected by a simple correction step to generate the new iterate. Since we will demonstrate the numerical superiority of Algorithm 1 via its comparison with the new prediction-correction method in Section 5, we here present the iterative scheme of the new prediction-correction method for the convenience of comparison. Recall that $v^k := (x_2^k, x_3^k, \cdots, x_m^k, \lambda^k)$.

Algorithm 2: a prediction-correction splitting method for solving (1.1) with $m \geq 3$:

Step 1. Prediction step. Let the iterate generated by Algorithm 1 from the given iterate $(x_1^k, x_2^k, \cdots, \lambda^k)$ be denoted by $(\tilde{x}_1^k, \tilde{x}_2^k, \cdots, \tilde{x}_m^k, \tilde{\lambda}^k)$.
Step 2. Correction step. Update the new iterate $(x_1^{k+1}, x_2^{k+1}, \cdots x_m^{k+1}, \lambda^{k+1})$ via:

$$\begin{cases} x_1^{k+1} = \tilde{x}_1^k; \\ v^{k+1} = v^k - \alpha_k(v^k - \tilde{v}^k), \end{cases} \tag{4.12}$$

where

$$\alpha_k = \gamma \alpha_k^*, \qquad \gamma_k \in (0, 2), \qquad \alpha_k^* = \frac{\varphi(v^k, \tilde{v}^k)}{\|v^k - \tilde{v}^k\|_G^2} \tag{4.13}$$

and

$$\varphi(v^k, \tilde{v}^k) = \|v^k - \tilde{v}^k\|_G^2 + (\lambda^k - \tilde{\lambda}^k)^T \big(\textstyle\sum_{i=2}^m A_i(x_i^k - \tilde{x}_i^k)\big). \tag{4.14}$$

As we can see easily, Algorithm 1 turns out to be a special case of Algorithm 2 where $\gamma_k \equiv 1/\alpha_k$ in (4.13). Thus, in the following, we prove the convergence for Algorithm 2, from which the convergence of Algorithm 1 becomes trivial.

Lemma 4.1 *Under the assumption* (3.5) *it holds that*

$$\varphi(v^k, \tilde{v}^k) \geq \frac{1+\tau}{2} \|v^k - \tilde{v}^k\|_G^2, \tag{4.15}$$

and consequently $\alpha_k^* > \frac{1+\tau}{2}$.

Proof. It follows from (3.5) that

$$2\|v^k - \tilde{v}^k\|_G^2 + 2(\lambda^k - \tilde{\lambda}^k)^T \big(\sum_{i=2}^m A_i(x_i^k - \tilde{x}_i^k)\big) \geq (1+\tau)\|v^k - \tilde{v}^k\|_G^2,$$

and the assertion follows from the definitions of $\varphi(v^k, \tilde{v}^k)$ and α_k^* (see (4.13) and (4.14)) directly. □

For the convenience of the later analysis, we summarize some equations in the follow lemma, whose proof is straightforward from (3.2) and (3.4) and thus omitted.

Lemma 4.2 *For the iterate generated by Algorithm 2, we have*

$$(\tilde{\lambda}^k - \lambda^{k+\frac{1}{2}}) = H\left(\textstyle\sum_{i=2}^m A_i(x_i^k - \tilde{x}_i^k)\right) \tag{4.16}$$

and

$$\sum_{i=1}^{m} A_i\tilde{x}_i^k - b = H^{-1}(\lambda^k - \tilde{\lambda}^k). \tag{4.17}$$

The following theorem justifies the rationale that we can use the descent step (4.12) as the correction step in Algorithm 2.

Theorem 4.3 *Let $\tilde{w}^k = (\tilde{x}_1^k, \tilde{v}^k)$ be generated by Algorithm 1, then we have*

$$(v^k - v^*)^T G(v^k - \tilde{v}^k) \geq \varphi(v^k, \tilde{v}^k), \quad \forall\, v^* \in \mathcal{V}^*, \tag{4.18}$$

where $\varphi(v^k, \tilde{v}^k)$ and G are defined in (4.14) *and* (3.6), *respectively.*

Proof. The proof consists of some manipulations. First, by deriving the optimal condition and considering (3.2), it is easy to see that (3.1) is characterized by the following VI:

$$\tilde{x}_1^k \in \mathcal{X}_1, \quad (x_1' - \tilde{x}_1^k)^T\big\{f_1(\tilde{x}_1^k) - A_1^T\lambda^{k+\frac{1}{2}}\big\} \geq 0, \quad \forall\, x_1' \in \mathcal{X}_1. \tag{4.19}$$

Similarly, the subproblems (3.3) can be characterized by the following VIs:

$$\tilde{x}_i^k \in \mathcal{X}_i, \ (x_i' - \tilde{x}_i^k)^T\big\{f_i(\tilde{x}_i^k) - A_i^T\lambda^{k+\frac{1}{2}} + \mu A_i^T H A_i(\tilde{x}_i^k - x_i^k)\big\} \geq 0, \ \forall\, x_i' \in \mathcal{X}_i, \ i = 2, \ldots, m. \tag{4.20}$$

Then, it follows from (4.19) and (4.20) that

$$\tilde{x}^k \in \mathcal{X}, \quad \begin{pmatrix} x_1' - \tilde{x}_1^k \\ x_2' - \tilde{x}_2^k \\ \vdots \\ x_m' - \tilde{x}_m^k \end{pmatrix}^T \begin{pmatrix} f_1(\tilde{x}_1^k) - A_1^T\lambda^{k+\frac{1}{2}} \\ f_2(\tilde{x}_2^k) - A_2^T\lambda^{k+\frac{1}{2}} \\ \vdots \\ f_m(\tilde{x}_m^k) - A_m^T\lambda^{k+\frac{1}{2}} \end{pmatrix}$$
$$\geq \begin{pmatrix} x_1' - \tilde{x}_1^k \\ x_2' - \tilde{x}_2^k \\ \vdots \\ x_m' - \tilde{x}_m^k \end{pmatrix}^T \begin{pmatrix} 0 \\ \mu A_2^T H A_2(x_2^k - \tilde{x}_2^k) \\ \vdots \\ \mu A_m^T H A_m(x_m^k - \tilde{x}_m^k) \end{pmatrix}, \quad \forall\, x' \in \mathcal{X}. \tag{4.21}$$

Because $\sum_{i=1}^m A_i\tilde{x}_i^k - b = H^{-1}(\lambda^k - \tilde{\lambda}^k)$ (see (4.17)), adding the equality

$$(\lambda' - \tilde{\lambda}^k)^T\big(\sum_{i=1}^{m} A_i\tilde{x}_i^k - b\big) = (\lambda' - \tilde{\lambda}^k)^T H^{-1}(\lambda^k - \tilde{\lambda}^k)$$

to (4.21), it yields

$$\tilde{w}^k \in \mathcal{W}, \quad \begin{pmatrix} x_1' - \tilde{x}_1^k \\ x_2' - \tilde{x}_2^k \\ \vdots \\ x_m' - \tilde{x}_m^k \\ \lambda' - \tilde{\lambda}^k \end{pmatrix}^T \left\{ \begin{pmatrix} f_1(\tilde{x}_1^k) - A_1^T\tilde{\lambda}^k \\ f_2(\tilde{x}_2^k) - A_2^T\tilde{\lambda}^k \\ \vdots \\ f_m(\tilde{x}_m^k) - A_m^T\tilde{\lambda}^k \\ \sum_{i=1}^m A_i\tilde{x}_i^k - b \end{pmatrix} + \begin{pmatrix} A_1^T(\tilde{\lambda}^k - \lambda^{k+\frac{1}{2}}) \\ A_2^T(\tilde{\lambda}^k - \lambda^{k+\frac{1}{2}}) \\ \vdots \\ A_m^T(\tilde{\lambda}^k - \lambda^{k+\frac{1}{2}}) \\ 0 \end{pmatrix} \right\}$$
$$\geq \begin{pmatrix} x_1' - \tilde{x}_1^k \\ x_2' - \tilde{x}_2^k \\ \vdots \\ x_m' - \tilde{x}_m^k \\ \lambda' - \tilde{\lambda}^k \end{pmatrix}^T \begin{pmatrix} 0 \\ \mu A_2^T H A_2(x_2^k - \tilde{x}_2^k) \\ \vdots \\ \mu A_m^T H A_m(x_m^k - \tilde{x}_m^k) \\ H^{-1}(\lambda^k - \tilde{\lambda}^k) \end{pmatrix}, \qquad \forall\, w' \in \mathcal{W}. \tag{4.22}$$

Set $w' = w^*$ in (4.22), use the notation of $F(\tilde{w}^k)$ and $(\tilde{w}^k - w^*)^T F(\tilde{w}^k) \ge 0$ (see (2.3) and (2.4)), we get

$$\begin{array}{rl} & \begin{pmatrix} \tilde{x}_1^k - x_1^* \\ \tilde{x}_2^k - x_2^* \\ \vdots \\ \tilde{x}_m^k - x_m^* \\ \tilde{\lambda}^k - \lambda^* \end{pmatrix}^T \begin{pmatrix} 0 \\ \mu A_2^T H A_2(x_2^k - \tilde{x}_2^k) \\ \vdots \\ \mu A_m^T H A_m(x_m^k - \tilde{x}_m^k) \\ H^{-1}(\lambda^k - \tilde{\lambda}^k) \end{pmatrix} \\ \ge & \big(\sum_{i=1}^m A_i(\tilde{x}_i^k - x_i^*)\big)^T(\tilde{\lambda}^k - \lambda^{k+\frac{1}{2}}), \qquad \forall w^* \in \mathcal{W}^*. \end{array} \tag{4.23}$$

By using $\sum_{i=1}^m A_i x_i^* = b$, (4.16) and (4.17), we have

$$\big(\sum_{i=1}^m A_i(\tilde{x}_i^k - x_i^*)\big)^T(\tilde{\lambda}^k - \lambda^{k+\frac{1}{2}}) = (\lambda^k - \tilde{\lambda}^k)^T\big(\sum_{i=2}^m A_i(x_i^k - \tilde{x}_i^k)\big).$$

Substituting it in the right hand side of (4.23), we obtain

$$\begin{array}{rl} & \begin{pmatrix} \tilde{x}_2^k - x_2^* \\ \vdots \\ \tilde{x}_m^k - x_m^* \\ \tilde{\lambda}^k - \lambda^* \end{pmatrix}^T \begin{pmatrix} \mu A_2^T H A_2(x_2^k - \tilde{x}_2^k) \\ \vdots \\ \mu A_m^T H A_m(x_m^k - \tilde{x}_m^k) \\ H^{-1}(\lambda^k - \tilde{\lambda}^k) \end{pmatrix} \\ \ge & (\lambda^k - \tilde{\lambda}^k)^T\big(\sum_{i=2}^m A_i(x_i^k - \tilde{x}_i^k)\big), \quad \forall v^* \in \mathcal{V}^*. \end{array} \tag{4.24}$$

Using the notations of G, v and $\varphi(v, \tilde{v})$, the assertion (4.18) follows from (4.24) immediately. □

Theorem 4.4 *Let $\tilde{w}^k = (\tilde{x}_1^k, \tilde{v}^k)$ be generated by Algorithm 1. Then for any $v^* \in \mathcal{V}^*$, we have*

$$\|\tilde{v}^k - v^*\|_G^2 \le \|v^k - v^*\|_G^2 - \big\{\|v^k - \tilde{v}^k\|_G^2 + 2(\lambda^k - \tilde{\lambda}^k)^T\big(\sum_{i=2}^m A_i(x_i^k - \tilde{x}_i^k)\big)\big\}. \tag{4.25}$$

Proof. By using (4.18) we obtain

$$\begin{array}{rl} & \|v^k - v^*\|_G^2 - \|\tilde{v}^k - v^*\|_G^2 \\ = & \|v^k - v^*\|_G^2 - \|v^k - v^* - (v^k - \tilde{v}^k)\|_G^2 \\ = & 2(v^k - v^*)^T G(v^k - \tilde{v}^k) - \|v^k - \tilde{v}^k\|_G^2 \\ \ge & \|v^k - \tilde{v}^k\|_G^2 + 2(\lambda^k - \tilde{\lambda}^k)^T\big(\sum_{i=2}^m A_i(x_i^k - \tilde{x}_i^k)\big). \end{array}$$

The assertion of this theorem is proved. □

Based on Theorem 4.4, convergence of the classical ALM (the special case of (1.1) with $m = 1$) and ADM (the special case of (1.1) with $m = 2$) can be immediately recovered, as we show in the following corollaries.

Corollary 4.5 *The sequence $\{\lambda^k\}$ generated by ALM for the problem* (1.1) *with $m = 1$ satisfies*

$$\|\lambda^{k+1} - \lambda^*\|_{H^{-1}}^2 \le \|\lambda^k - \lambda^*\|_{H^{-1}}^2 - \|\lambda^k - \lambda^{k+1}\|_{H^{-1}}^2, \ \ \forall \lambda^* \in \Lambda^*. \tag{4.26}$$

Proof. The ALM is the special case of the proposed method for (1.1) with $m = 1$. In this case, $v = \lambda$ and (4.25) becomes (4.26). The assertion is proved. $\square$

Note that the ALM is essentially the application of the well-known proximal point algorithm [36] to the Lagrange dual of (1.1), as analyzed insightfully in [41]. If $H = \beta I$ is a scalar matrix, then the inequality (4.26) is reduced to

$$\|\lambda^{k+1} - \lambda^*\|^2 \le \|\lambda^k - \lambda^*\|^2 - \|\lambda^k - \lambda^{k+1}\|^2, \ \ \forall\, \lambda^* \in \Lambda^*,$$

which reflects this fact clearly.

Corollary 4.6 *The sequence* $\{v^k = (x_2^k, \lambda^k)\}$ *generated by ADM for the problem* (1.1) *with* $m = 2$ *(i.e.,* (1.2)*) satisfies*

$$\|v^{k+1} - v^*\|_G^2 \le \|v^k - v^*\|_G^2 - \|v^k - v^{k+1}\|_G^2, \quad \forall\, v^* \in \mathcal{V}^*. \tag{4.27}$$

Proof. Note that ADM for the problem (1.2) is exactly the application of Algorithm 1 for (1.1) with $m = 2$ and $\mu = 1$. In this case,

$$G = \begin{pmatrix} A_2^T H A_2 & \\ & H^{-1} \end{pmatrix}$$

By setting $m = 2$ and $v^{k+1} = \tilde{v}^k$, it follows from (4.25) that

$$\|v^{k+1} - v^*\|_G^2 \le \|v^k - v^*\|_G^2 - \big(\|v^k - v^{k+1}\|_G^2 + 2(\lambda^k - \lambda^{k+1})^T A_2 (x_2^k - x_2^{k+1})\big).$$

Therefore, in order to show (4.27), we need only to prove

$$(x_2^k - x_2^{k+1})^T A_2^T (\lambda^k - \lambda^{k+1}) \ge 0. \tag{4.28}$$

Since $\mu = 1$, it follows from (4.20) that

$$x_2^{k+1} \in \mathcal{X}_2, \ \ (x_2' - x_2^{k+1})^T \big\{ f_2(x_2^{k+1}) - A_2^T \lambda^{k+\frac{1}{2}} + A_2^T H A_2 (x_2^{k+1} - x_2^k) \big\} \ge 0, \ \ \forall\, x_2' \in \mathcal{X}_2. \tag{4.29}$$

Note that in the case that $m = 2$, by using (3.2) and (3.4), we have

$$\lambda^{k+1} = \lambda^{k+\frac{1}{2}} + HA(x_2^k - \tilde{x}_2^k).$$

Thus, (4.29) can be rewritten as

$$x_2^{k+1} \in \mathcal{X}_2, \ \ (x_2' - x_2^{k+1})^T \big\{ f_2(x_2^{k+1}) - A_2^T \lambda^{k+1} \big\} \ge 0, \ \ \forall\, x_2' \in \mathcal{X}_2. \tag{4.30}$$

Due to the same reason, in the $(k-1)$-th iteration, we have

$$x_2^k \in \mathcal{X}_2, \ \ (x_2' - x_2^k)^T \big\{ f_2(x_2^k) - A_2^T \lambda^k \big\} \ge 0, \ \ \forall\, x_2' \in \mathcal{X}_2. \tag{4.31}$$

Setting $x_2' = x^k$ in (4.30) and $x_2' = x^{k+1}$ in (4.31), respectively, and then adding the resulting inequalities, we get

$$(x_2^k - x_2^{k+1})^T A_2^T (\lambda^k - \lambda^{k+1}) \ge (x_2^k - x_2^{k+1})^T \big(f_2(x_2^k) - f_2(x_2^{k+1}) \big).$$

Because f_2 is monotone, (4.28) is true and thus the assertion is proved $\square$

Now, we are ready to prove the convergence of the proposed methods.

Theorem 4.7 *Let $\tilde{w}^k = (\tilde{x}_1^k, \tilde{v}^k)$ be generated by Algorithm 1 and the parameter μ be chosen such that the condition* (3.5) *is satisfied. If the new iterate is updated by $v^{k+1} = \tilde{v}^k$, then we have*

$$\|v^{k+1} - v^*\|_G^2 \le \|v^k - v^*\|_G^2 - \tau\|v^k - \tilde{v}^k\|_G^2, \ \ \forall\, v^* \in \mathcal{V}^*. \tag{4.32}$$

Proof. It follows from Theorem 4.4 and the assumption (3.5) immediately. □

In order to explain how to determine the step size α_k in (4.13), we define the step-size-dependent new iterate by

$$v(\alpha) = v^k - \alpha(v^k - \tilde{v}^k). \tag{4.33}$$

In this way,

$$\vartheta(\alpha) = \|v^k - v^*\|_G^2 - \|v(\alpha) - v^*\|_G^2 \tag{4.34}$$

is the distance decrease functions in the k-th iteration by using updating form (4.33). By defining

$$q(\alpha) = 2\alpha\varphi(v^k, \tilde{v}^k) - \alpha^2\|v^k - \tilde{v}^k\|_G^2. \tag{4.35}$$

It follows from (4.33), (4.34) and (4.18) that

$$\begin{aligned} \vartheta(\alpha) &= \|v^k - v^*\|_G^2 - \|v^k - v^* - \alpha(v^k - \tilde{v}^k)\|_G^2 \\ &= 2\alpha(v^k - v^*)^T G(v^k - \tilde{v}^k) - \alpha^2\|v^k - \tilde{v}^k\|_G^2 \\ &\ge 2\alpha\varphi(v^k, \tilde{v}^k) - \alpha^2\|v^k - \tilde{v}^k\|_G^2 \\ &= q(\alpha). \end{aligned} \tag{4.36}$$

Note that $q(\alpha)$ is a quadratic function of α, it reaches its maximum at

$$\alpha_k^* = \frac{\varphi(v^k, \tilde{v}^k)}{\|v^k - \tilde{v}^k\|_G^2}, \tag{4.37}$$

and this is just the same as defined in (4.13). Usually, in practical computation, taking a relaxed factor $\gamma > 1$ is useful for fast convergence.

Theorem 4.8 *Let $\tilde{w}^k = (\tilde{x}_1^k, \tilde{v}^k)$ be generated by Algorithm 1 and the parameter μ be chosen such that the condition* (3.5) *is satisfied. If the new iterate v^{k+1} is updated by* (4.12)-(4.14) *in Algorithm 2, then we have*

$$\|v^{k+1} - v^*\|_G^2 \le \|v^k - v^*\|_G^2 - \frac{\gamma(2-\gamma)(1+\tau)^2}{4}\|v^k - \tilde{v}^k\|_G^2, \quad \forall\, v^* \in \mathcal{V}^*. \tag{4.38}$$

Proof. It follows from (4.34) and (4.36) that

$$\|v^{k+1} - v^*\|_G^2 \le \|v^k - v^*\|_G^2 - q(\gamma\alpha_k^*), \quad \forall v^* \in \mathcal{V}^*. \tag{4.39}$$

By using (4.35) and (4.37) we obtain

$$\begin{aligned} q(\gamma\alpha_k^*) &= 2\gamma\alpha_k^*\varphi(v^k, \tilde{v}^k) - (\gamma\alpha_k^*)^2\|v^k - \tilde{v}^k\|_G^2 \\ &= \gamma(2-\gamma)\alpha_k^*\varphi(v^k, \tilde{v}^k). \end{aligned} \tag{4.40}$$

Using the results in Lemma 4.1 we have

$$\alpha_k^*\varphi(v^k, \tilde{v}^k) \ge \frac{(1+\tau)^2}{4}\|v^k - \tilde{v}^k\|_G^2.$$

The proof of this theorem is complete. □

In order to guarantee that the right hand side of (4.40) is positive, we take $\gamma \in [1, 2)$. Theorems 4.7 and 4.8 point out that the sequence $\{v^k\}$ generated by the proposed methods is Fejér monotone with respect to $\mathcal{V}^*$. Now, we are in the stage to prove the main convergence theorem of this paper.

Theorem 4.9 *Let $\{v^k\}$ be generated by Algorithm 2. Then we have*

1. $lim_{k\to\infty}\|A_i(x_i^k-\tilde{x}_i^k)\|=0,\ \ i=2,\dots,m,$ *and* $\lim_{k\to\infty}\|\lambda^k-\tilde{\lambda}^k\|=0.$
2. *Any cluster point of $\{\tilde{w}^k\}$ is a solution point of VI$(\mathcal{W},F)$.*
3. *The sequence $\{\tilde{v}^k\}$ converges to some $v^\infty\in\mathcal{V}^*$ when $A_i, i=1,\dots,m,$ are full column rank matrices.*

Proof. First, it follows from (4.32) and(4.38) that

$$\min\Big\{\frac{\gamma(2-\gamma)(1+\tau)^2}{4},\tau\Big\}\cdot\sum_{k=0}^{\infty}\|v^k-\tilde{v}^k\|_G^2\le\|v^0-v^*\|_G^2,\quad \forall\, v^*\in\mathcal{V}^*.$$

Consequently, since H is positive definite, we have

$$\lim_{k\to\infty}\|A_i(x_i^k-\tilde{x}_i^k)\|=0,\ \ i=2,\dots,m,\quad \text{and}\quad \lim_{k\to\infty}\|\lambda^k-\tilde{\lambda}^k\|=0. \tag{4.41}$$

The first assertion is proved. Substituting (4.41) in (4.21), we have

$$\lim_{k\to\infty}(x_i'-\tilde{x}_i^k)^T\big(f_i(\tilde{x}_i^k)-A_i^T\lambda^{k+\frac{1}{2}}\big)\ge0,\quad \forall x_i'\in\mathcal{X}_i,\ \ i=1,\dots,m.$$

Because (see (4.16)), we have

$$\lim_{k\to\infty}(\tilde{\lambda}^k-\lambda^{k+\frac{1}{2}})=\lim_{k\to\infty}H\big(\textstyle\sum_{i=2}^m A_i(x_i^k-\tilde{x}_i^k)\big)=0$$

and thus

$$\lim_{k\to\infty}(x_i'-\tilde{x}_i^k)^T\big(f_i(\tilde{x}_i^k)-A_i^T\tilde{\lambda}^k\big)\ge0,\quad \forall x_i'\in\mathcal{X}_i,\ \ i=1,\dots,m. \tag{4.42}$$

Note that $\lim_{k\to\infty}\|\lambda^k-\tilde{\lambda}^k\|=0$ implies (see (3.4)) that that

$$\lim_{k\to\infty}(\sum_{i=1}^m A_i\tilde{x}_i^k-b)=0. \tag{4.43}$$

Combining (4.42) and (4.43) we get

$$\tilde{w}^k\in\mathcal{W},\quad \lim_{k\to\infty}(w-\tilde{w}^k)^TF(\tilde{w}^k)\ge0,\quad \forall\, w\in\mathcal{W}, \tag{4.44}$$

and thus any cluster point of $\{\tilde{w}^k\}$ is a solution point of VI$(\mathcal{W},F)$.

In the case that $A_i, i=2,\dots,m$ are column full rank matrices, the matrix G is positive definite. It follows from (4.32) and (4.38) that $\{v^k\}$ is bounded and

$$\lim_{k\to\infty}\|v^k-\tilde{v}^k\|=0.$$

Let v^∞ be a cluster point of $\{\tilde{v}^k\}$ and the subsequence $\{\tilde{v}^{k_j}\}$ converges to v^∞. It follows from (4.44) that

$$\tilde{w}^{k_j}\in\mathcal{W},\quad \lim_{k\to\infty}(w-\tilde{w}^{k_j})^TF(\tilde{w}^{k_j})\ge0,\quad \forall\, w\in\mathcal{W}. \tag{4.45}$$

This means that $v^\infty\in\mathcal{V}^*$. Since $\{v^k\}$ is Fejér monotone and $\lim_{k\to\infty}\|v^k-\tilde{v}^k\|=0$, the sequence $\{\tilde{v}^k\}$ can not have other cluster point and $\{\tilde{v}^k\}$ converges to $v^\infty\in\mathcal{V}^*$. $\Box$

5 Applications

In this section, we show the superiority and efficiency of the proposed Algorithm 1 by some concrete applications of (1.1). All the codes were written by MATLAB 7.8 (R2009a) and were run on a T6500 notebook with the Intel Core 2 Duo CPU at 2.1 GHz and 2 GB of memory.

5.1 The superiority of Algorithm 1 without correction steps

In this subsection, we mainly justify the necessity of presenting such an algorithm without correction steps for solving (1.1), as we have mentioned in Introduction. More specifically, we will focus on the model of recovering low-rank and sparse components of matrices from incomplete and noisy observation which was recently proposed in [48] based on the pioneering work [6, 8] (see also [35, 55]). We will implement the proposed Algorithm 1 and the alternating directions based contraction (ADBC) method in [29] to solve this matrix recovery model, and then the rational of developing Algorithm 1 without any correction steps for (1.1) is clear.

The task of recovering the low-rank and sparse components of a given matrix captures many applications arising in diversified areas such as the model selection in statistics, matrix rigidity in computer science and system identification in engineering. In [8], the following convex relaxation model of this NP-hard problem was presented:

$$\begin{array}{rl} \min_{A,E} & \|A\|_* + \tau\|E\|_1 \\ \text{s.t.} & A + E = C, \end{array} \tag{5.1}$$

where $C \in \mathcal{R}^{m\times n}$ is the given matrix (data); the nuclear norm denoted by $\|\cdot\|_*$ is to induce the low-rank component of C and the l_1 norm denoted by $\|\cdot\|_1$ is to induce the sparse component of C; and $\tau > 0$ is a constant balancing the low-rank and sparsity. This model has also been highlighted in [6] in the context of the so-called robust principle component analysis (RPCA) where C is a given high-dimensional matrix in $\mathcal{R}^{m\times n}$, A is the underlying low-rank matrix representing the principal components and E is the corrupted data matrix which is sparse yet its entries can be arbitrary in magnitude.

The model (5.1) was immediately extended in [48] to more general setting where only a fraction of entries of the matrix are assumed to be observable and the observation is corrupted by both impulsive and Gaussian noise. The resulted model is:

$$\begin{array}{rl} \min_{A,E} & \|A\|_* + \tau\|E\|_1 \\ \text{s.t.} & \|P_\Omega(C - A - E)\|_F \le \delta, \end{array} \tag{5.2}$$

where Ω is a subset of the index set of entries $\{1, 2, \cdots, m\} \times \{1, 2, \cdots, n\}$ representing the observable entries; $P_\Omega : \mathcal{R}^{m\times n} \to \mathcal{R}^{m\times n}$ summarizes the incomplete observation information and it is denoted by the orthogonal projection onto the span of matrices vanishing outside of Ω so that the ij-th entry of $P_\Omega(X)$ is X_{ij} if $(i, j) \in \Omega$ and zero otherwise; $\delta > 0$ is the Gaussian noise level and $\|\cdot\|_F$ is the standard Frobenius norm.

As in [48], we let $C = A^* + E^*$ be the data matrix, where A^* and E^* are the low-rank and sparse components to be recovered, respectively. We generate A^* by $A = LR^T$, where L and R are independent $m \times r$ matrices whose elements are i.i.d. Gaussian random variables with zero means and unit variance. Hence, the rank of A^* is `r`. The index of observed entries, i.e. Ω, is determined at random. The support $\Gamma \subset \Omega$ of the impulsive noise E^* (sparse but large) are chosen uniformly at random, and the non-zero entries of E^* are i.i.d. uniformly in the interval $[-500, 500]$. Let `sr`, spr and rr represent the ratios of sample (observed) entries (i.e., $|\Omega|/mn$), the number of non-zero entries of E (i.e., $\|E\|_0/mn$) and the rank of A^* (i.e., r/m), respectively.

To witness the fact that correction steps are likely to destroy the low-rank characteristic of the recovered low-rank component, we focus on the special case of (5.2) with

$$\delta = 0, \quad \tau = 1/\sqrt{n}, \quad m = n = 500, \quad \texttt{rr} = \texttt{spr} = 0.05 \text{ and } \texttt{sr} = 0.6.$$

We executed the singular value decomposition (SVD) fully to compute the exact ranks of the recovered low-rank components A^k, and recorded the variation of the ranks for Algorithm 1 and the ADBC method in the following figure (left).

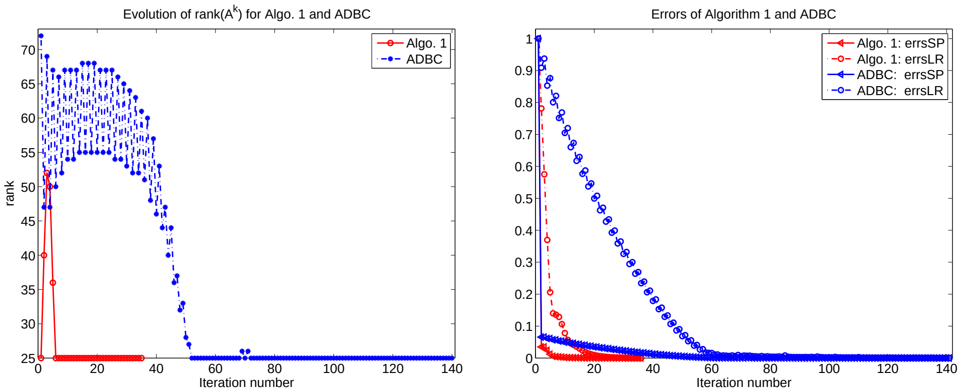

Figure 1: Evolution of rank (left) and errors of recovered components (right) for Algorithm 1 and ADBC

As shown in Figure 1, the rank of iterates generated by ADBC changes radically according to iterations at the first stage; while the rank of iterates generated by Algorithm 1 is much more stable—not sensitive to the iterations. We believe that the reason behind this difference is that the correction step of ADBC destroys the underlying low-rank feature. On the opposite, the low-rank feature is well preserved by Algorithm 1; and this merit is very suitable for the application of some popular packages for partial SVD such as PROPACK.

For exposing the difference resulted by correction steps, we also compare their respective variations of the recovered low-rank component's error and sparse component's error denoted by

$$errsLR := \frac{\|A^k - A^*\|_F}{\|A^*\|_F} \quad \text{and} \quad errsSP := \frac{\|E^k - E^*\|_F}{\|E^*\|_F}$$

in the right of Figure 1.

As shown clearly, compared to Algorithm 1, the errors generated by ADBC's iterations, especially the recovered low-rank component's error, change more radically.

Therefore, Figure 1 illustrates well our motivation of presenting such an splitting algorithm without any correction steps for solving (1.1).

5.2 The efficiency of Algorithm 1

In this subsection, we mainly show the efficiency of the proposed Algorithm 1 to solve some applications of (1.1) arising in the area of image processing. We also implement the proposed Algorithm 2 in this subsection. With the comparison of Algorithms 1 and 2, the reason of avoiding correction steps is further justified.

As delineated in [26], the model (1.1) captures many important applications of digital image restoration and reconstruction problems arising in image processing. Digital image restoration and reconstruction problems play an important role in various areas of applied sciences such as medical and astronomical imaging, film restoration and image and video coding and many others. Let $\bar{x} \in \mathcal{R}^{n^2}$ be the original image, $K \in \mathcal{R}^{n^2 \times n^2}$ be a blurring (or convolution) operator, $S \in \mathcal{R}^{n^2 \times n^2}$ be a diagonal matrix, which diagonal entry is 0 (i.e. missing the corresponding information) or 1 (i.e. keeping the corresponding information), $\omega \in \mathcal{R}^{n^2}$ be an additive noise, and $f \in \mathcal{R}^{n^2}$ which satisfies the relationship

$$f = SK\bar{x} + \omega. \tag{5.3}$$

Given S, K, our objective is to recover $\bar{x}$ from f, which is known as the problem of image inpainting. When S is the identity operator, recovering $\bar{x}$ from f is referred to as deconvolution.

It is well known that (5.3) is an ill-conditioned problem. To stabilize the recovery, one must utilize some prior information such as adding a regularizer to certain data fidelity. The total variation (TV) regularization introduced in [43] for image construction has been shown both experimentally and theoretically to be suitable for preserving sharp edges. More specifically, denote the discrete gradient operators as $\partial_1 : \mathcal{R}^{n^2} \to \mathcal{R}^{n^2}$ and $\partial_2 : \mathcal{R}^{n^2} \to \mathcal{R}^{n^2}$, which represent the discretized derivatives in the horizontal and vertical directions, respectively. The gradient operator is then defined as $\nabla := (\partial_1, \partial_2)$. Then the TV inpainting model can be formulated as the following reconstruction model

$$\min_x \||\nabla x|\| + \frac{\mu}{2}\|SKx - f\|_N, \tag{5.4}$$

or the equivalent constrained form:

$$\begin{array}{ll} \min & \||\nabla x|\| \\ s.t. & x \in \mathcal{R}^{n^2},\ \|SKx - f\|_N \le \alpha. \end{array} \tag{5.5}$$

Here, let $y = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \in \mathcal{R}^{n^2} \times \mathcal{R}^{n^2}$, we define $|y| = \sqrt{y_1^2 + y_2^2} \in \mathcal{R}^{n^2}$, and $\| \cdot \|_N$ refers a norm, in particular $\| \cdot \|_2 = \| \cdot \|$. The parameter α is positive real numbers, which measures the trade-off between the fit to f and the amount of regularization.

In this following, we apply both the proposed methods to solve the constrained TV-l^2 model, i.e., let $N = 2$ in (5.5), which is to restore a blurred image with additive Gaussian noise. To ensure the nonempty of the solution set, we assume that the constraint set

$$\Omega := \{x \in \mathcal{R}^{n^2},\ \|SKx - f\| \le \alpha\} \tag{5.6}$$

is nonempty.

First, we rewrite the problem (5.5) into:

$$\begin{array}{ll} \min & \||y|\| \\ s.t. & y = \nabla x, \\ & Kx = z, \\ & z \in \mathcal{Z}, \end{array} \tag{5.7}$$

where

$$\mathcal{Z} := \{z \in \mathcal{R}^{n^2},\ \|Sz - f\| \le \alpha\}$$

is a ball. Then, (5.7) is a special case of (1.1) where

$$x = (x_1, x_2, x_3) = (y, x, z) \in \mathcal{R}^{2n^2} \times \mathcal{R}^{n^2} \times \mathcal{Z}$$

$$f_1(y) = |||y|||, f_2(x) = 0, f_3(z) = 0$$

and

$$A = (A_1, A_2, A_3) = \begin{pmatrix} I & -\nabla & 0 \\ 0 & -K & I \end{pmatrix} \quad \text{and } b = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

The augmented Lagrangian functional associated to this problem is defined by

$$\mathcal{L}(x, y, z, \lambda_1, \lambda_2) = |||y|||_1 - \lambda_1^T(y - \nabla x) - \lambda_2^T(z - Kx) + \frac{1}{2}\|y - \nabla x\|_H^2 + \frac{1}{2}\|z - Kx\|_H^2.$$

To expose the main idea clear, throughout we set $H \equiv \beta I$ and we restrict out discussion into the case where $\beta > 0$ is fixed.

Now, we elaborate on the strategy of solving the resulted subproblems when the proposed Algorithm 1 is applied to solve (5.7). Recall that we denote by $(\tilde{y}^k, \tilde{\lambda}^k, \tilde{x}^k), \tilde{z}^k$ the iterate generated by Algorithm 1 for solving (5.7).

- The first subproblem (3.1) amounts to solving

$$\tilde{y}^k \in \operatorname{argmin}_y |||y|||_1 + \frac{\beta}{2}\|y - \nabla x^k - \frac{1}{\beta}\lambda_1^k\|^2.$$

  Obviously, the closed-form solution of the above minimization problem is given by

$$\tilde{y}^k = \text{shrink}_{\frac{1}{\beta}}\left(\nabla x^k + \frac{\lambda_1^k}{\beta}\right),$$

  where for a vector $y = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \in \mathcal{R}^{n^2} \times \mathcal{R}^{n^2}$ and a scalar $\beta > 0$, the operator $\text{shrink}_\beta(y)$ is defined as

$$\text{shrink}_\beta(y) = y - \min(\beta, |y|) \cdot \frac{y}{|y|}, \tag{5.8}$$

  where $|y| \in \mathcal{R}^{n^2}$ be defined by $|y| = \sqrt{y_1^2 + y_2^2}$ and $0 \cdot (0/0) = 0$ is assumed.

- The second subproblem (3.2) is trivial whenever $\tilde{y}^k$ is computed.

- The third subproblem (3.3) amounts to solving

$$\tilde{x}^k \in \operatorname{argmin}_x (\lambda_1^{k+\frac{1}{2}})^T \nabla x + (\lambda_2^{k+\frac{1}{2}})^T (Kx) + \frac{\beta}{2}\mu\| \begin{pmatrix} -\nabla \\ -K \end{pmatrix} (x - x^k)\|^2,$$

  whose solution can be obtained via solving the following system of linear equations:

$$\beta\mu(\nabla^T\nabla + K^T K)(\tilde{x}^k - x^k) = -\nabla^T \lambda_1^{k+\frac{1}{2}} - K^T \lambda_2^{k+\frac{1}{2}}. \tag{5.9}$$

  Assuming that the blur K is spatially invariant and periodic boundary conditions are used for the discrete differential operator, we see that the matrices K and $\nabla^T\nabla$ can be diagonalized by the Fast Fourier transform (FFT).

- The fourth subproblem (3.4) amounts to solving

$$\tilde{z}^k \in \operatorname{argmin}_z (-\lambda_2^{k+\frac{1}{2}})^T z + \frac{\beta}{2}\mu\|z - z^k\|^2,$$

  whose closed-form solution is given by:

$$\tilde{z}^k = P_B[z^k + \frac{1}{\beta\mu}\lambda^{k+\frac{1}{2}}].$$

  Here the projection operator $P_B(\cdot)$ is defined as

$$P_B(t) = \frac{t-f}{\|t-f\|} .* \min(\|t-f\|, \alpha) + f,$$

  where $t \in \mathcal{R}^{n^2}$.

In the following, we implement both Algorithms 1 and 2 to the Lena image (256×256). The iteration will be terminated whenever

$$\frac{\|x^{k+1} - x^k\|}{\max\{\|x^k\|, 1\}} < \epsilon, \tag{5.10}$$

where $\epsilon > 0$ is a given tolerance. As usually used, we measure the quality of restoration by the signal-to-noise ratio (SNR), which is measured in decibel (dB) and defined by

$$\mathrm{SNR}(x) \triangleq 10 * \log_{10} \frac{\|\bar{x} - \tilde{x}\|^2}{\|\bar{x} - x\|^2}, \tag{5.11}$$

where $\bar{x}$ is the original image and $\tilde{x}$ is the mean intensity value of $\bar{x}$. In the following experiments, Guassian kernel was applied with blurring size `hsize` $= 5$, standard deviation $\sigma = 14$ under two cases: one is noise level `std` $= 10^{-3}$ with 40% information, the other is noise level `std` $= 10^{-2}$ with 30% information. We set the missing pixels as 0. We set $\epsilon = 5 \times 10^{-3}$, $\alpha = \|Kx - f\|$, and $\mu = 1.8$, $\gamma = 1.3$, $\beta = 18$. In Figure 2, we present the destroyed image and recover results from two methods in terms of relative error, SNR, consuming time (CPU) and iteration numbers (It.).

6 Conclusions

For solving the separate convex programming problem with linking linear constraints whose objective function is in the form of the sum of m individual functions without crossed variables, we present the first splitting algorithm where m smaller and easier subproblems are solved separately at each iteration and no any correction steps are required. The superiority and efficiency of the new method is illustrated by some numerical examples. At each iteration of the new method, however, the resulted m subproblems are not suitable for completely simultaneous computation as the latter $m - 1$ subproblems require the solution of the first subproblem. With the consideration of using advanced computing infrastructure such as parallel facilities, it is of interest to develop such a splitting algorithm whose resulted subproblems at each iteration are completely tailored for simultaneous computation. This is one of our research topics in the future.

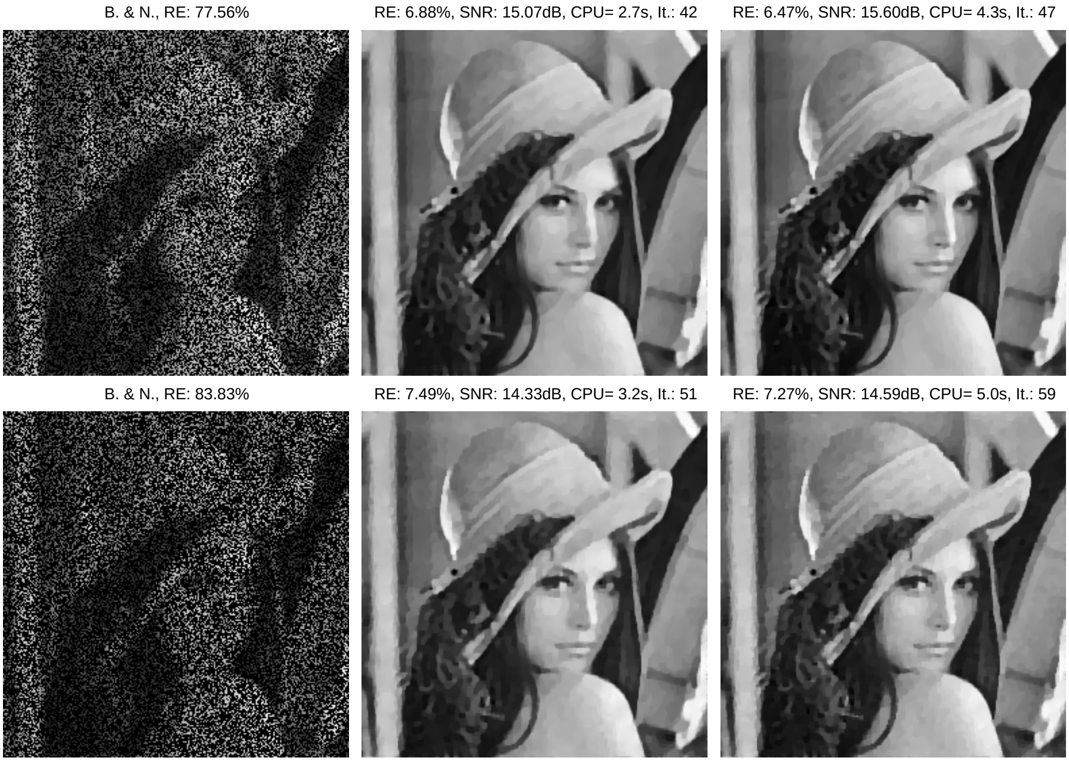

Figure 2: Recovered results from Algo. 1 and 2 on TV-l^2. In both rows from left to right: blurry and noisy image missing partial information (B. & N.) and recovered results from Algorithm 1. and 2., respectively. RE, CPU and It. represent relative error, running time and iteration numbers, respectively. Top row: $\texttt{std} = 10^{-3}$ and lost 60% information; Bottom row: $\texttt{std} = 10^{-2}$ and missing 70% information.

References

[1] M. V. Afonso, J. M. Bioucas-Dias and M. A. T. Figueiredo, Fast image recovery using variable splitting and constrained optimization, manuscript, 2009.

[2] A. Auslender and M. Teboulle, Entropic proximal decomposition methods for convex programs and variational inequalities, Math. Program., 91, 33-47, 2001.

[3] D. P. Bertsekas, *Constrained Optimization and Lagrange Multiplier methods*, Academic Press, 1982.

[4] E. Blum and W. Oettli. *Mathematische Optimierung.* Springer-Verlag, Berlin, 1975. Grundlagen und Verfahren, Mit einem Anhang "Bibliographie zur Nichtlinearer Programmierung", Ökonometrie und Unternehmensforschung, No. XX.

[5] N. Bose and K. Boo, High-resolution image reconstruction with multisensors, International Journal of Imaging Systems and Technology, 9, 294-304, 1998.

[6] E. J. Candès, X. Li, Y. Ma and J. Wright, Robust Principal Component Analysis, manuscript, 2009.

[7] Y. Censor, T. Elfving, N., Kopf and T. Bortfeld, The multiple-sets split feasibility problem and its applications for inverse problems, Inverse Problems, 21, 2071-2084, 2005.

[8] V. Chandrasekaran, S. Sanghavi, P. A. Parrilo and A. S. Willskyc, Rank-sparsity incoherence for matrix decomposition, manuscript, http://arxiv.org/abs/0906.2220.

[9] G. Chen and M. Teboulle, A proximal-based decomposition method for convex minimization problems, *Mathematical Programming*, 64, 81-101, 1994.

[10] P. L. Combettes and J.-C. Pesquet, Proximal splitting methods in signal processing, in: Fixed-Point Algorithms for Inverse Problems in Science and Engineering, (H. H. Bauschke, R. Burachik, P. L. Combettes, V. Elser, D. R. Luke, H. Wolkowicz, Editors), New York: Springer-Verlag, 2010.

[11] J. Douglas and H. H. Rachford, On the numerical solution of the heat conduction problem in 2 and 3 space variables, *Transactions of the American Mathematical Society*, 82, 421-439, 1956.

[12] J. Eckstein, Some saddle-function splitting methods for convex programming, *Optimization Methods and Software*, 4, 75-83, 1994.

[13] J. Eckstein and D. P. Bertsekas, On the Douglas-Rachford splitting method and the proximal points algorithm for maximal monotone operators, Math. Program., 55, 293-318., 1992.

[14] J. Eckstein and B. F. Svaiter, General projective splitting methods for sums of maximal monotone operators, SIAM J. Control Optim., 48, 787-811, 2009.

[15] E. Esser, Applications of Lagrangian-Based alternating direction methods and connections to split Bregman, UCLA CAM Report 09-31, 2009.

[16] J. Eckstein and M. Fukushima, Some reformulation and applications of the alternating direction method of multipliers, *Large Scale Optimization: State of the Art*, W. W. Hager *et al* eds., Kluwer Academic Publishers, pp. 115-134, 1994.

[17] F. Facchinei and J.-S. Pang, Finite-Dimensional Variational Inequalities and Complementarity problems, Volume I, Springer Series in Operations Research, Springer-Verlag, 2003.

[18] M. Fukushima, Application of the alternating direction method of multipliers to separable convex programming problems, *Computational Optimization and Applications*, 2, pp. 93-111, 1992.

[19] D. Gabay and B. Mercier, A dual algorithm for the solution of nonlinear variational problems via finite-element approximations, *Computer and Mathematics with Applications*, 2, pp. 17-40, 1976.

[20] D. Gabay, Applications of the method of multipliers to variational inequalities, *Augmented Lagrange Methods: Applications to the Solution of Boundary-valued Problems*, M. Fortin and R. Glowinski, eds., North Holland, Amsterdam, The Netherlands, pp. 299-331, 1983.

[21] R. Glowinski, *Numerical Methods for Nonlinear Variational Problems*, Springer-Verlag, New York, Berlin, Heidelberg, Tokyo, 1984.

[22] R. Glowinski and P. Le Tallec, *Augmented Lagrangian and Operator-Splitting Methods in Nonlinear Mechanics*, SIAM Studies in Applied Mathematics, Philadelphia, PA, 1989.

[23] D. Goldfarb and S. Q. Ma, Fast alternating linearization methods for minimizing the sum of two convex functions, manuscript, 2009.

[24] D. Goldfarb and S. Q. Ma, Fast multiple splitting algorithms for convex optimization, manuscript, 2009.

[25] T. Goldstein and S. Osher, The split Bregman method for L1-regularized problems, SIAM J. Imaging Sciences, 2 (2), pp. 323-343, 2009.

[26] D. R. Han, X. M. Yuan and W. X. Zhang, An augmented-Lagrangian-based parallel splitting method for linearly constrained separate convex programming with applications to image processing, submission, 2010.

[27] B. S. He, Parallel splitting augmented Lagrangian methods for monotone structured variational inequalities, *Computational Optimization and Applications*, 42, 195-212, 2009.

[28] B. S. He, L. Z. Liao, D. Han and H. Yang, A new inexact alternating directions method for monontone variational inequalities, *Mathematical Programming*, 92, 103-118, 2002.

[29] B. S. He, M. Tao, M. H. Xu and X. M. Yuan, Alternating directions based contraction method for generally separable linearly constrained convex programming problems, *Manuscript*, 2009.

[30] B. S. He, M. H. Xu and X. M. Yuan, Solving large-scale least squares covariance matrix problems by alternating direction methods, SIAM J. Matrix Analysis Appli., in revision.

[31] M. Hestenes, Multiplier and gradient methods, J. Opti. Theory Appli., 4, 303-320, 1969.

[32] K. C. Kiwiel, C. H. Rosa and A. Ruszczyński, Proximal decomposition via alternating linearization, SIAM J. Opti., 9(3), 668-689, 1999.

[33] S. Kontogiorgis and R. R. Meyer, A variable-penalty alternating directions method for convex optimization, *Mathematical Programming*, 83, 29-53, 1998.

[34] Z. K. Jiang and X. M. Yuan, A new parallel descent-like method for solving a class of variational inequalities, *Journal of Optimization Theory and Applications*, 145(2), 311-323, 2010.

[35] Z.-C. Lin, M.-M. Chen, L.-Q. Wu, Y. Ma, The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrices, manuscript, 2009.

[36] B. Martinet, Regularision d'inéquations variationnelles par approximations successive, Revue Francaise d'Automatique et Informatique Recherche Opérationnelle, 126, 154-159, 1970.

[37] M. Ng, F. Wang and X. M. Yuan, Fast minimization methods for solving constrained total-variation superresolution image reconstruction, Multidimensional Systems and Signal Processing, to appear.

[38] M. Ng, P. A. Weiss and X. M. Yuan, Solving constrained total-variation problems via alternating direction methods, SIAM J. Sci. Comput., to appear.

[39] J. Nocedal and S. J. Wright, *Numerical Optimization.* Second Edition, Spriger Verlag, 2006.

[40] M. Powell, A method for nonlinear constraints in minimization problems, in Optimization, R. Fletcher (Editor), pp. 283-298, Academic Press, New York, 1969.

[41] R. T. Rockafellar, Monotone operators and the proximal point algorithm, SIAM J. Control Optim., 126, 877-898, 1976.

[42] C. H. Rosa, Pathways of Economic Development in an Uncertain Environment: A Finite Scenario Approach to the U.S. Region under Carbon Emission Restrictions, WP-94-41, International Institute for Applied Systems Analysis, Laxenburg, Austria, 1994.

[43] L. Rudin, S. Osher and E. Fatemi, Nonlinear total variation based noise removal algorithms, Physica D, 60, 259-268, 1992.

[44] S. Setzer, Split Bregman algorithm, Douglas-Rachford splitting, and frame shrinkage, Lecture Notes in Computer Science, 5567, pp. 464-476, 2009.

[45] S. Setzer, G. Steidl and T. Tebuber, Deblurring Poissonian images by split Bregman techniques, J. Visual Communication and Image Representation, 21, pp. 193-199, 2010.

[46] J. E. Spingarn, Partial inverse of a monotone operator, Appl. Math. Optim., 10, 247-265, 1983.

[47] J. Sun and S. Zhang, A modified alternating direction method for convex quadratically constrained quadratic semidefinite programs, manuscript, 2009.

[48] M. Tao and X. M. Yuan, Recovering low-rank and sparse components of matrices from incomplete and noisy observations, SIAM J. Optim, in revision.

[49] P. Tseng, Alternating projection-proximal methods for convex programming and variational inequalities, SIAM J. Optim., 7, 951-965, 1997.

[50] Z. Wen, D. Goldfarb and W. Yin, Alternating direction augmented Lagrangian methods for semidefinite programming, manuscript, 2009.

[51] Z. Wen, D. Goldfarb, A line search multigrid method for large-scale convex optimization, manuscript, 2007.

[52] J. Yang and Y. Zhang, Alternating direction algorithms for l_1-problems in compressive sensing, TR09-37, CAAM, Rice University.

[53] C. H. Ye and X. M. Yuan, A descent method for structured monotone variational inequalities, Optim. Methods Softw., 22, 329-338, 2007.

[54] X. M. Yuan, Alternating direction methods for sparse covariance selection, manuscript, 2009.

[55] X. M. Yuan and J. Yang, Sparse and low rank sparse matrix decomposition via alternating directions method, manuscript, 2009.